Dipl.-Ing. Erich Kobler, BSc

On Learning Regularizers for Inverse Problems in Imaging

DOCTORAL THESIS

to achieve the university degree of

Doktor der technischen Wissenschaften

submitted to

Graz University of Technology

Supervisor

Thomas Pock, Univ.-Prof. Dr.techn. Dipl.-Ing.

Institute of Computer Graphics and Vision
Graz University of Technology, Austria

Ozan Öktem, Assoc.-Prof. PhD.

Department of Mathematics
KTH - Royal Institute of Technology, Sweden

Graz, November 2020

To my family, Silke and Rosalie

Truth . . . is much too complicated to allow anything but approximations.

– John von Neumann (1947)

AFFIDAVIT

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present doctoral thesis.

Date, Signature

Abstract

Numerous problems in computer vision and medical imaging can be cast as inverse problems. The variational approach robustly estimates solutions of an inverse problem by minimizing an energy composed of a data fidelity term and a regularizer. While the data fidelity term is utilized to incorporate knowledge about the underlying physical process of the inverse problem, the regularizer typically encodes a-priori statistical properties of the desired solutions. Classically, handcrafted regularizers motivated by first-order statistics of images are used, which are frequently outperformed by state-of-the-art purely data-driven deep learning models. In this thesis, we develop novel methods combining variational methods and deep learning that lead to state-of-the-art results on various imaging tasks and allow a rigorous mathematical analysis.

First, we introduce variational networks (VNs) to explore links between well-studied incremental variational methods and deep learning — specifically residual neural networks (ResNets). In its core, a VN consists of several parametric incremental proximal gradient steps, which can be flexibly adapted to learn proximal gradient schemes, incremental proximal gradient schemes, and ResNet-like models. The flexibility of VNs allows us to study the limitations of convexity for these schemes in the context of image restoration. Our numerical results on image denoising and deblurring suggest that variational models utilizing convex regularizers cannot represent all aspects of natural images and are outperformed by non-convex regularizers. In contrast, we can improve the results for both convex and non-convex regularizers by facilitating parametric incremental proximal methods. In all experiments, we observe the phenomenon that the different VN types require only a few steps to yield reasonable results.

We further investigate the effect that in variational methods the best image quality is frequently observed when the associated gradient flow is stopped before converging to a stationary point. We argue that this phenomenon originates from a tradeoff between optimization and modeling errors and remains valid even if highly expressive deep learning-based regularizers are employed. We analyze this paradox by considering a variational method featuring a parametric regularizer and by introducing an optimal stopping time in the corresponding gradient flow. This optimal stopping time as well as the parameters of the regularizer are determined by a mean-field optimal control approach, where the gradient flow defines the state equation. Moreover, we propose a novel data-driven general-purpose regularizer called total deep variation (TDV), which exploits recent architectural design patterns from deep learning to overcome the limited expressiveness of the fields of experts (FoE) regularizer advocated in VNs. The TDV regularizer is a convolutional neural network (CNN) that extracts local features on multiple scales and in successive blocks to assign an energy to every image pixel. The combination of the mean-field optimal control training problem and the highly expressive TDV regularizer leads to state-of-the-art results on various image restoration and reconstruction problems and simultaneously enables a rigorous mathematical analysis. We prove the existence of solutions of the mean-field optimal control problem in the time-continuous and time-discrete setting and characterize the stability with respect to initial value and parameter variations. Finally, we experimentally verify the robustness against adversarial attacks and numerically derive upper bounds for the generalization error.

Keywords. Convolutional neural networks, gradient flow, image restoration, inverse problems, mean-field optimal control problem, medical imaging, variational methods, variational networks

Kurzfassung

Zahlreiche Probleme in der Bildverarbeitung und der medizinischen Bildgebung können als inverse Probleme formuliert werden. Variationsmethoden bieten die Möglichkeit die Lösungen eines inversen Problems robust zu schätzen, indem sie eine Energie minimieren, die aus einem Datenterm und einem Regularisierer besteht. Während der Datenterm verwendet wird um Wissen über den zugrunde liegenden physikalischen Prozess eines inversen Problems zu modellieren, codiert der Regularisierer typischerweise statistische Eigenschaften der gewünschten Lösungen. Klassischerweise werden manuell modellierte Regularisierer verwendet, die auf Gradientenstatistiken von natürlichen Bildern basieren. Diese werden häufig von rein datengesteuerten Deep-Learning-Modellen übertroffen. In dieser Doktorarbeit entwickeln wir neue Methoden, die Variationsmethoden und Deep Learning kombinieren und zu herausragenden Ergebnissen bei verschiedenen Problemen in der Bildverarbeitung führen und eine detaillierte mathematische Analyse ermöglichen.

Zuerst definieren wir VNs um Zusammenhänge zwischen inkrementellen Variationsmethoden und Deep Learning und im Speziellen ResNets zu untersuchen. In ihrem Kern bestehen VNs aus mehreren parametrischen inkrementellen proximalen Gradientenschritten, die flexibel angepasst werden können, um proximale Gradientenschemata, inkrementelle proximale Gradientenschemata und ResNet-ähnliche Modelle zu lernen. Die Flexibilität von VNs ermöglicht es uns, die Grenzen der Konvexität für diese Schemata im Kontext der Bildverarbeitung zu untersuchen. Unsere numerischen Ergebnisse für das Entrauschen und Entzerren von Bildern mit Variationsmodellen legen nahe, dass konvexe Regularisierer nicht alle Aspekte natürlicher Bilder darstellen können und von nicht konvexen Regularisierern übertroffen werden. Im Gegensatz dazu können die Ergebnisse sowohl für konvexe als auch für nicht konvexe Regularisierer verbessert werden, indem parametrische inkrementelle proximale Methoden verwendet werden. In all unseren Experimenten beobachten wir das Phänomen, dass die verschiedenen VN-Typen nur sehr wenige Schritte für vernünftige Ergebnisse benötigen.

Darüber hinaus untersuchen wir den Effekt, dass bei Variationsmethoden häufig die beste Bildqualität beobachtet wird, wenn der zugehörige Gradientenfluss gestoppt wird, bevor er zu einem stationären Punkt konvergiert. Wir argumentieren, dass dieses Phänomen auf einem Kompromiss aus Optimierungs- und Modellierungsfehlern beruht und auch dann gültig bleibt, wenn auf Deep Learning basierende Regularisierer verwendet werden. Wir analysieren dieses Paradoxon, indem wir Variationsmethoden mit parametrischen Regularisierern betrachten und eine optimale Stoppzeit in den entsprechenden Gradientenfluss einführen. Die optimale Stoppzeit sowie die Parameter des Regularisierers werden durch ein Mean-field optimales Steuerungsproblem bestimmt, bei dem der Gradientenfluss die Zustandsgleichung definiert. Außdem schlagen wir einen neuartigen datengesteuerten Allzweck-Regularisierer namens TDV vor, der die jüngsten architektonischen Entwurfsmuster von Deep Learning nutzt, um die begrenzte Ausdruckskraft des in VNs verwendeten FoE Regularisierers zu überwinden. Der TDV Regularisierer ist ein CNN, das lokale Merkmale auf mehreren Skalen und in aufeinanderfolgenden Blöcken extrahiert, um jedem Bildpunkt eine Energie zuzuweisen. Die Kombination des Mean-field optimalen Steuerungsproblems und des ausdrucksstarken TDV Regularisierers führt zu Ergebnissen auf dem neuesten Stand der Technik bei zahlreichen Problemen in der Bildverarbeitung und ermöglicht gleichzeitig eine genaue mathematische Analyse. Unter anderem zeigen wir die Existenz von Lösungen des Mean-field optimalen Steuerungsproblems in der zeitkontinuierlichen und zeitdiskreten Variante und charakterisieren die Stabilität bezüglich Anfangswert- und Parametervariationen. Schließlich verifizieren wir experimentell die Robustheit gegenüber 'adversarial' Angriffen und leiten numerisch Obergrenzen für den Generalisierungsfehler ab.

Acknowledgments

Doing a PhD is quite a long voyage with many ups and downs. In the next lines, I would like to take the chance and express my gratitude to all colleagues, friends, and relatives that enabled this extraordinary endeavor.

In the fall of 2015, Tom encouraged me to participate in the PhD regatta in his team and match myself with fellow researches around the world. Tom, I am very grateful for this opportunity, the excellent equipment and environment you provided, and your trust in me and my skills. I would like to thank you for your motivation and patience, the countless hours that we spent discussing on various topics, the freedom to explore the scientific world, and pointing out new directions when I got lost. Your constant pursuit of new methods to achieve your goals has been truly inspiring and highly contagious. You are an outstanding researcher and the best advisor one could possibly imagine. I would also like to thank Ozan for agreeing to be my second supervisor.

Of course maneuvering a yacht requires an experienced crew. When I first set sails to explore the scientific field, I had no itinerary and only vague ideas about where to go. I am truly thankful to all team members of the vision learning and optimization group who had open ears for any question, taught me scientific practice, pointed me towards interesting papers and ideas, eagerly engaged in endless discussions, and made me feel like home at the office. Kerstin, Teresa, Gottfried, Sasha, Christian R, Christoph, Yura, Patrick, Thomas G, Markus, Joana, Alexander, Dominik, Lea, Thomas P, Robert, Christian K, and Lydia thank you all for that. Additionally, I would like to thank Gottfried and Joana for sharing an office with me and turning it into a pleasant workspace. In particular, I would like to thank Kerstin and Teresa with whom I spent most of my work time in the first two years. I still remember the countless hours we spent implementing our deep learning framework for variational networks with a python frontend and C++ backend that supported some form of automated differentiation. It was big fun. I really enjoyed our social team building activities and the weekly running program helped to keep me focused this summer. I never thought that I would run a half marathon on a Monday evening just for fun. Moreover, I am really glad that I met Alexander at a workshop in Münster in 2017. First, we were strangers but soon after we became colleagues and friends while we were working on a learning-based image morphing model. Later Alexander joined our team in Graz and turned into a close friend. Thank you Alexander for all the nights we spent coding and writing papers in the office discussing the formulation of every sentence. Being with you in the last two years has been a real pleasure.

I am also truly thankful for the opportunities that have been opening up along this journey. I could travel around the world to present and discuss my research at various venues from Canada to Vietnam. Further, a big thank you to Ricardo for taking me as a research intern at the Memorial Sloan Kettering cancer center in New York. It was a great experience to collaborate with new faces from different fields in this vibrant city.

Embarking on such a challenging voyage is easier from a save haven. I am truly grateful that my wonderful family is such a place for me — I love you all. Thank you daddy for your kind heart, open mind, and unconditional support. Thank you Reinmar for being at my side since our birth. You are the best brother and friend one could possibly wish for. Finally, I would like to express my greatest thanks to my wife Silke. Since the beginning of my PhD studies, you have always been there for me despite the countless long nights and weekends, and you helped me to stay grounded in stressful situations. Silke, you gave me the best present — our daughter Rosalie. I love you both so much.

Contents

List of Figures

List of Tables

1 Introduction

Various applications in imaging target the robust estimation of unknown scalar- or vector-valued images given noisy and indirect observations. These so-called inverse problems are typically hard to solve because the observations often do not cover all relevant aspects of the unknown image, and small modeling, numerical, or measurement errors frequently lead to heavily corrupted results. Variational methods address these issues by solving approximate problems that utilize prior knowledge about the structure of solutions. These solutions are characterized as minimizers of an energy typically composed of a data fidelity term and a regularizer. While the data fidelity term models the application-specific knowledge of the acquisition process, the regularizer is used to incorporate knowledge about the regularity or statistical properties of the desired solution. In the last decades, variational methods have been used to tackle numerous computer vision and medical imaging problems such as denoising [1], deblurring [2], segmentation [3], optical flow [4], positron emission tomography [5], single photon emission computed tomography [6], and magnetic resonance imaging (MRI) reconstruction [7] due to their simplicity and solid theoretical foundations. The recent success of deep learning [8] in numerous fields has led to a paradigm shift in the computer vision and imaging community, though. The research focus has shifted from modeling appropriate data fidelity and regularization terms towards developing suitable parametric functions in the form of networks that can be adapted to solve specific tasks by adjusting their parameters. The underlying idea is that these networks learn a meaningful solution strategy by extracting relevant statistical information from training data, which can be transferred to later process unseen instances of the same problem. In this thesis, we establish connections between both approaches and combine variational methods and deep learning to get the best of both worlds — theoretically well-understood models and state-of-the-art numerical results.

In the following sections, we first define inverse problems and discuss some prototypical examples in imaging. Then, the variational approach to robustly estimate their solutions is explained and various recent data-driven extensions are reviewed. Finally, we conclude this chapter by stating the contributions and the outline of this thesis.

1.1 Inverse Problems in Imaging

Numerous applications and scientific disciplines rely on the stable estimation of an unknown quantity by observing and measuring its effects. This identification task of the unknown quantity given possibly noisy and incomplete observations is frequently called an *inverse problem*. The associated *forward problem* ("predict the observations given an estimate of the unknown quantity") is often simple to solve and is determined by the underlying physical process. In contrast, inverse problems are typically hard to solve and ill-posed according to Hadamard [9], which means that one of the following conditions does not hold:

- The solution exists.
- The solution is unique.
- The solution continuously depends on the observations.

Thus, the solutions of ill-posed inverse problems are usually sensitive to measurement errors, i.e. small errors in the measurements may induce large errors in the resulting

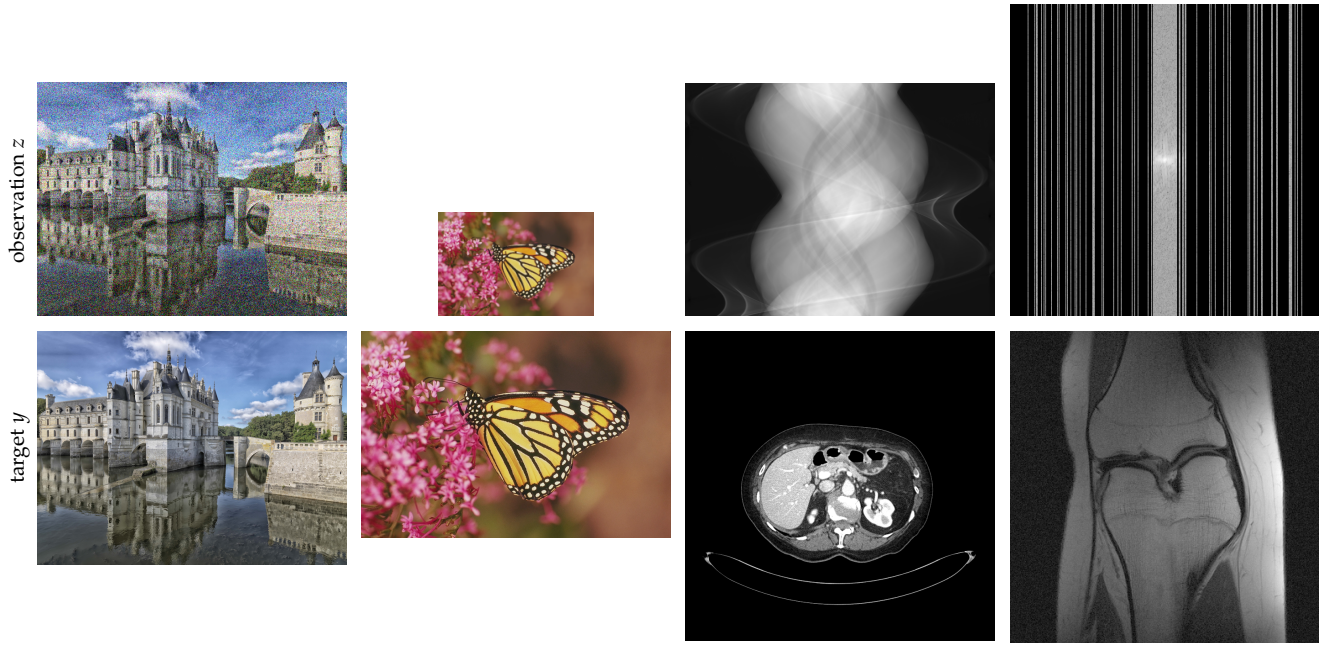


Figure 1.1: Illustration of the observation (first row) and desired target (second row) of four prototypical linear inverse problems in imaging. First column: image denoising for a sample from the DIV2K dataset [11] corrupted by 20% Gaussian noise. Second column: the monarch image of the Set14 dataset (bottom) along with the 2-fold downsampling result. Third column: a sample from the Mayo dataset [12] (bottom) along with an 8-fold angular undersampled sinogram (top). Fourth column: a knee image from the fastMRI dataset [13] (bottom) and the corresponding 4-fold undersampled k-space data (top).

estimate of the unknown quantity, or multiple estimates explain the observations equally well. Therefore, it is important to account for the ill-posed nature of inverse problems to enable a reliable decision process upon their solutions. For instance, medical diagnosis frequently relies on imaging techniques such as X-ray, computed tomography (CT), or MRI that are based on solving inverse problems. Likewise, various applications in computer vision such as localization and autonomous driving heavily depend on depth or motion estimation, which also define inverse problems [10].

Many inverse problems in digital imaging relate the unknown *target* $y \in \mathcal{Y}$, which may represent a discrete image, video, volume or vector field, and the possibly noisy *observation* $z \in \mathcal{Z}$ utilizing the *linear* forward model

$$z = Ay + \xi. \tag{1.1}$$

Here, $\mathcal{Y}$ and $\mathcal{Z}$ are finite-dimensional vector spaces and the linear *forward operator* $A : \mathcal{Y} \to \mathcal{Z}$ describes the physical process mapping a target to the corresponding noise-free observation. The measurement noise is represented by $\xi \in \Xi$, where Ξ is also a finite-dimensional vector space. Throughout this thesis, we assume that we have detailed knowledge about the forward operator A and the distribution $\mathcal{T}_\Xi$ of the measurement noise ξ. Then, inverse problems determine the target y that best explains the measurement z. In the following, we present some prototypical inverse problems.

In digital image processing, we often encounter image *restoration* problems, which aim at estimating a target image y given a degraded image z. One of the simplest image restoration problems is *image denoising*. Here, the observed image z is equal to the target image subject to additive noise $\xi \sim \mathcal{T}_\Xi$, as illustrated in the first column of Figure 1.1. Thus, the forward operator in (1.1) is equal to the identity mapping $A = \mathrm{Id}$ and the target image space coincides with the degraded image space, i.e. $\mathcal{Y} = \mathcal{Z}$. The noise ξ is in the easiest case assumed to be independent and identically distributed (i.i.d.) Gaussian ($\xi \sim \mathcal{N}(0, \sigma^2 \mathrm{Id})$). However, this assumption is often too restrictive. Realistic noise is frequently a mixture of Gaussian and Poisson noise [14], which

can be approximated by a heteroskedastic Gaussian distribution. This is a spatially varying Gaussian distribution whose standard deviation is a function of the unknown target image y.

A different restoration task in image processing is *single image super-resolution (SISR)*. In this task, we would like to increase the resolution of a low-resolution image $z \in \mathcal{X} \subset \mathbb{R}^l$ by a factor $s \in \mathbb{N}$. The linear forward model typically relates the high-resolution target image $y \in \mathcal{Y} \subset \mathbb{R}^{ls^2}$ with the observation by an s-fold downsampling operation that involves a 2-dimensional interpolation kernel. Thus, the linear operator A in (1.1) implements a convolution with the interpolation kernel utilizing a stride of length s. A sample pair of the SISR task consisting of a low-resolution observation and a high-resolution target is depicted in the second column of Figure 1.1 using a super-resolution factor $s = 2$. There are many possible solutions for the SISR inverse problem since high-frequency information is lost due to the downsampling operation. Moreover, the higher the downsampling stride is, the less high-frequency information is available in the low-resolution image z, which results in a more complicated restoration problem. SISR restoration problems with stride $s = 1$, i.e. $\mathcal{Y} = \mathcal{X}$, are also called *deconvolution* problems if the interpolation kernel is known. If the interpolation kernel is in addition unknown, the resulting problems are called *blind deconvolution* problems.

Various inverse problems in medical imaging are *reconstruction* problems such as computed tomography (CT) or magnetic resonance imaging (MRI) reconstruction. Here, the target space $\mathcal{Y}$ and observation space $\mathcal{X}$ do not coincide, in contrast to restoration problems. The task of CT is to reconstruct a map whose elements reflect the X-ray attenuation of the underlying matter from a sequence of X-ray measurements acquired around the scanned object. These X-ray projections are aggregated in the so-called sinogram. The sinogram of a sample scan is depicted at the top in the third column of Figure 1.1, while the corresponding attenuation map is shown at the bottom. The elements of the sinogram can be computed by integrating the attenuation map in the area spanned by the X-ray source and the corresponding detector element. Hence, the linear operator A in (1.1) approximates many area integrals to relate the attenuation map with the sinogram measurements. To account for the different noise sources in the acquisition process, the noise distribution of ξ is typically modeled as a heteroskedastic Gaussian [15, 16]. In medical CT, we aim at reducing the X-ray dose exposed to patients during a CT scan while maintaining the reconstruction quality. There are different dose reduction techniques, which we discuss in detail in Section 5.2.2 and Section 6.6.

Another important imaging technique in medical imaging is MRI. A receiver coil in an MRI scanner measures complex-valued k-space data z, whose elements reflect changes in the magnetism of atomic nuclei excited by different frequencies [17]. Thus, each k-space entry represents a Fourier coefficient of the imaged volume and the linear forward operator in (1.1) is given by the discrete 2-dimensional Fourier transform. To reduce the typically rather long acquisition time of MRI scans, compressed sensing (CS) [18] and parallel imaging techniques [19, 20] are frequently adopted. In this case, a scanner simultaneously acquires incomplete k-space data of multiple receiver coils that are sensitive in overlapping areas of the imaged object. Accelerated clinical MRI scans often utilize a Cartesian undersampling to acquire only a fraction of the k-space data columns or rows. The incomplete k-space data of a receiver coil used in a four times accelerated parallel scan is illustrated in the fourth column of Figure 1.1 along with its target image. Clearly, there are multiple possible images that explain the observed k-space data equally well since a large fraction of the data is missing. As in the SISR image restoration task, higher undersampling results in a more ambiguous inverse problem.

In this section, we have demonstrated that many practically relevant tasks in computer vision and medical imaging are inverse problems. The discussed problems are just a small subset of inverse problems arising in imaging. Further inverse problems are, for

instance, optical flow estimation [4, 10], ultrasound imaging [21], or positron emission tomography [5].

1.2 From Variational Methods to Data-driven Models

In the following, we provide a short overview of solution approaches to inverse problems starting from classical variational methods to data-driven models. First, we define variational methods and elaborate on their statistical interpretation. Then, we briefly review classical handcrafted regularizers and provide a summary of parametric regularizers and associated parameter identification methods in image processing. A short discussion of fully-learned reconstruction models motivated by deep learning concludes this section.

To account for the ill-posed nature of inverse problems, numerous *regularization* techniques evolved from the pioneering work of Tikhonov [22] and Phillips [23]. The fundamental idea of regularization techniques is to transform an ill-posed inverse problem into an approximate well-posed problem [24], whose unique solution exists and continuously depends on the observation [9]. Hence, Tikhonov [22] proposed to compute approximate solutions of a linear inverse problem by solving the variational problem

$$\min_{x\in\mathcal{X}} \tfrac{1}{2}\left\|Ax - z\right\|^2 + \tfrac{\lambda}{2}\left\|x\right\|^2,$$

where the first term penalizes the quadratic deviation of the forward model Ax from the measurement $z \in \mathcal{Z}$, the second term enforces the regularity of the solution $x \in \mathcal{X}$, and $\lambda > 0$ is a scalar balancing parameter. The underlying idea of Tikhonov regularization is to avoid degenerated solutions by ensuring that its solution is bounded. This penalization approach was extended to *variational methods* [25, 26], which aim at solving variational problems of the form

$$\inf_{x\in\mathcal{X}} \left\{\mathrm{E}(x, z) \coloneqq \mathrm{D}(x, z) + \mathrm{R}(x)\right\}. \tag{1.2}$$

Here, the *data fidelity term* $\mathrm{D} : \mathcal{X} \times \mathcal{Z} \to \mathbb{R}$ typically uses the forward operator A to penalize deviations of the noise-free observation Ax from the measurement $z \in \mathcal{Z}$, while the *regularizer* $\mathrm{R} : \mathcal{X} \to \mathbb{R}$ imposes knowledge about the solution by penalizing undesired properties. The *energy* $\mathrm{E} : \mathcal{X} \times \mathcal{Z} \to \mathbb{R}$ combines both terms and determines a scalar value that characterizes the quality of the solution. In the case of Tikhonov regularization, the regularizer $\mathrm{R}(x) = \frac{\lambda}{2}\left\|x\right\|^2$ implies that the solution has a bounded norm, i.e. $\|x\| < \infty$. However, this penalization of the amplitude of the solution is not suitable for many image processing tasks since it favors dark images. A very popular regularizer avoiding this issue is the *total variation (TV)*, which assumes that the gradient of images has small variation and is sparse. It was introduced to the image processing community by Rudin, Osher, and Fatemi in their seminal paper [1]. The TV regularizer became a driving force in the development of modern regularization techniques in imaging [24] and was extended in various ways [27–30]. The flexible choice of the regularizer and the data fidelity term is a particular advantage of variational methods that enables the formulation of reconstruction problems specifically tailored for the application at hand.

There is also a statistical interpretation that links variational methods with probability theory [31]. This statistical viewpoint provides a rigorous framework to account for uncertainties that arise in inverse problems due to measurement errors or loss of information. According to Bayes' theorem, the posterior probability density $p(x|z)$ of an estimate $x \in \mathcal{X}$ that approximates the unknown target $y \in \mathcal{Y}$ given the observation $z \in \mathcal{Z}$ following (1.1) is defined as

$$p(x|z) = \frac{p(z|x)p(x)}{\int_{\mathcal{X}} p(z|x')p(x')\,\mathrm{d}x'},$$

where $p(z|x)$ is the *data likelihood* and $p(x)$ the *prior*. The data likelihood $p(z|x)$ specifies how well a solution x explains the observed data z and is often characterized by the forward model (1.1) and the probability density of the noise ξ. In contrast, the prior $p(x)$ encodes all knowledge about the structure of a solution and thus represents the belief in the solution itself. As a result, the belief in a certain solution x given the observations z is quantified by the posterior probability. Usually, we are interested in the solution that maximizes the posterior probability, which is known as the maximum a-posteriori (MAP) estimator [32]

$$\widehat{x} \in \operatorname*{argmax}_{x \in \mathfrak{X}} p(x|z).$$

Taking the negative logarithm results in the equivalent problem

$$\widehat{x} \in \operatorname*{argmin}_{x \in \mathfrak{X}} -\log p(z|x) - \log p(x).$$

If we compare this variational minimization problem with the variational formulation of inverse problems (1.2), we see that the data fidelity term D can be identified with the negative data log-likelihood $-\log p(z|x)$ and the regularizer R with the negative log-prior $-\log p(x)$. Thus, regularizers allow to incorporate statistical knowledge about the solution into variational methods.

One of the earliest works that combine the variational approach with the statistical modeling of image properties is due to Geman and Geman [33]. They related image restoration problems with finding low-energy states of a configuration of gray-values over a discrete pixel grid determined by a Markov random field (MRF) [34]. Their MRF formulation defines an energy consisting of a data likelihood and a prior term that penalizes the differences of local pixel neighbors using a potential function. Motivated by the statistics of natural images, Geman and McClure [6] advocated non-convex potential functions that match the high kurtosis and long exponential tail distribution of local pixel differences [35, 36]. Driven by results from robust statistics numerous potential functions have been proposed to regularize pixel differences [6, 37–43]. It became apparent that natural image priors should incorporate higher-order information of larger pixel neighborhoods. However, modeling the statistical relations of higher-order pixel neighborhoods is challenging due to their complexity [43, 44], and the large dimensionality of images. Therefore, Zhu and Mumford suggested learning regularizers from natural image samples [45, 46].

The need to incorporate higher-order statistical information of natural images into regularization models throve the development of parametric regularizers whose parameters are identified by learning from data. Inspired by the patch-based product of experts machine learning model of Hinton et al. [47–49], Roth and Black [50, 51] proposed the fields of experts (FoE) regularizer. It is one of the most successful parametric regularizer models in imaging and can be considered as a generalization of the TV regularizer. The FoE regularizer uses several pairs of convolution filters and simple parametric potential functions to assign a local regularization energy to every pixel. The sum of the pixel-wise energy defines the FoE regularizer.

In the last decade, various learning approaches have been proposed that determine the parameters of the FoE regularizer from data. While the FoE regularizer was originally trained in a generative way using contrastive divergence [48], Samuel and Tappen [52] and Chen et al. [53] showed that discriminative learning using bilevel optimization to adapt the regularization parameters such that the MAP estimator is close to a target leads to a performance increase. To avoid solving the computationally intensive lower-level problems in bilevel optimization, Domke [54] proposed to unroll a gradient scheme to compute approximate MAP estimates using only a fixed number of iteration steps. Allowing the parameters of the FoE regularizer to additionally adapt in each iteration step results in trainable nonlinear reaction diffusion (TNRD)

models [55], which can be interpreted as incremental gradient methods and are a special case of variational networks (VNs) [56].

To rigorously analyze such truncated iterative schemes, we framed the training process as an optimal control problem of a gradient flow with finite time horizon using the FoE regularizer [57]. Later, we formulated the learning of regularizer parameters by early stopping gradient flows as a mean-field optimal control problem [58] and proposed a deep learning inspired regularizer called total deep variation (TDV) [59, 60]. This combination enabled a detailed analysis of the learned FoE and TDV regularizers by means of stochastic stability bounds, nonlinear eigenfunctions and worst-case estimates. Moreover, the truncated iterative training approach in conjunction with the TDV regularizer leads to state-of-the-art results on numerous imaging tasks.

Recently, further approaches were introduced that determine the parameters of deep learning motivated regularizers in a discriminative way. Lunz et al. [61] proposed to learn the parameters of a deep learning inspired, feed-forward regularizer such that its energy discriminates the distributions of target samples and corrupted samples motivated by the Wasserstein formulation of generative adversarial networks (GANs) [62]. In contrast, Li et al. [63] defined a data-driven regularizer by applying a coercive potential function to the features of a convolutional neural network (CNN). Here, the CNN is the encoding part of an encoder-decoder CNN that is trained to predict the residual error, i.e. the difference of (non)-corrupted input images and their associated target images. All the aforementioned regularizers maintain a variational structure, i.e. they characterize the image quality by an energy, which can be used in gradient-based algorithms to estimate the solution.

An alternative approach to combine iterative methods and deep learning is to directly learn a proximal reconstruction scheme. One of the first methods is due to Gregor and LeCun [64]. They observed that iterative algorithms resemble recurrent neural networks (RNNs) and proposed to learn all parameters of the iterative shrinkage and thresholding algorithm including the step sizes and the linear operators from data, which resulted in computationally more effective sparse coding schemes. This idea was also transferred to other first-order proximal methods. Vogel and Pock [65] proposed to learn the step sizes, convolution operators and the thresholds of nonlinear point-wise proximal operators of an unrolled primal-dual hybrid gradient (PDHG) method [66] with iteration-dependent parameters inspired by TNRD [55]. This idea was extended by the learned primal-dual networks of Adler and Öktem [67]. They suggested to replace the primal and dual proximal gradient steps in the PDHG method by iteration dependent CNNs that operate on a history of primal/dual variables and use the forward operator and its adjoint to link the primal and dual sequence. The resulting feed-forward network defines a mapping from the observation and an initial estimate to its output reconstruction.

In parallel, so-called plug-and-play priors [68] or regularization by denoising [69] methods have been developed. The fundamental idea of these approaches is to replace the proximal mapping in an iterative proximal algorithm such as PDHG [66] or alternating direction method of multipliers (ADMM) [70, 71] by an existing denoising algorithm. Initially, classical denoising methods such as non-local means [72] or BM3D [73] were advocated to replace the proximal mapping. Later, these ideas were combined with deep learning-based denoising methods [74, 75]. For a more detailed evaluation of recent data-driven models in imaging, we refer the interested reader to the excellent review papers [76, 77].

The recent success of deep learning methods in the field of inverse problems in imaging was driven by three major effects. First, large and diverse datasets [13, 78] provide enough samples to approximate the distribution of real-world problems well. Second, there has been an incredible boost in the computing power of graphics processing units (GPUs), which are the backbone of modern deep learning methods. The high-performance computing power of Nvidia's GPUs increased by a factor of 9

within the last four years[1], which enables faster training and inference. Moreover, the available memory on GPUs strongly increased, which has paved the way to learn deeper and larger models also for memory-intensive inverse problems such as 3-dimensional CT reconstruction, where the raw measurements alone require giga bytes (GBs) of memory. Finally, machine learning and in particular deep learning methods are easily accessible due to well documented, open source frameworks such as TensorFlow [79] and PyTorch [80]. These frameworks simplify the development of machine learning models by providing fundamental tools such as data loading and preprocessing, automatic differentiation, and various optimization algorithms.

1: This is the geometric mean over different high-performance computing applications according to `https://bit.ly/2G527S4` accessed September, 2020.

1.3 Contribution and Outline

Since 2015, the field of data-driven methods designed to solve inverse problems has strongly evolved. At that time the TNRD models of Chen and Pock [55] yielded state-of-the-art results on many image restoration tasks and we were interested in developing its theoretical foundations. Therefore, we proposed to view the parametric TNRD models as incremental methods [81] that perform proximal gradient steps on a cyclic sequence of parametric energy functions, which resulted in variational networks (VNs) [56]. It turned out that also residual neural networks (ResNets) [82] can be interpreted as incremental proximal methods provided that the residual functions are gradients. Using the VN framework, we investigated the role of convexity in the context of parametric energy functions for image denoising and deblurring. In addition, we applied VNs to reconstruct low-dose 3-dimensional helical CT clinical scans and compared tube-current dose reduction with SparseCT [83]. Later, we formulated the training of truncated parametric gradient schemes as an optimal control problem of time-continuous gradient flows including the stopping time in the trainable parameters [57]. Here, we proved the existence of solutions of the optimal control problem and derived first- and second-order optimality conditions for the stopping time using the FoE regularizer. To overcome the limited expressiveness of the FoE regularizer, we proposed the total deep variation (TDV) regularizer [59] designed by recent deep learning principles, which resulted in state-of-the-art results on various image restoration and reconstruction tasks. To determine the parameters of the TDV regularizer and the stopping time of the gradient flow, we phrased the training problem as a mean-field optimal control problem [60], which enables a rigorous mathematical analysis. We showed the existence of solutions of the time-continuous and time-discrete optimal control problem in the mean-field setting using a semi-implicit discretization scheme. Moreover, we developed stochastic upper bounds for the stability w.r.t. input and parameter variations of the learned gradient scheme. Additionally, a nonlinear eigenfunction analysis of the learned parametric regularizers enables insights in their local behavior. Finally, we numerically validated stability bounds for the proposed optimal control problem and estimated an empirical upper bound for the generalization error.

The remainder of this thesis is structured as follows:

Chapter 2 provides a brief overview of mathematical concepts and results from functional analysis, measure theory, probability theory, ordinary differential equations, and optimization that are used throughout the thesis. Then, we discuss basic learning types, elaborate on the difference between generative and discriminative learning, and define the concept of generalization in the context of machine learning in Chapter 3. In addition, we define neural networks (NNs) in Chapter 3 and illustrate recent architectural design patterns successfully applied in deep learning.

In Chapter 4, we first analyze the statistics of natural image gradients and 2×2 image patches and later review classical regularizers motivated by image statistics. Then, we present the parametric FoE and TDV regularizers and discuss different parameter estimation approaches. Chapter 5 is devoted to VNs, which establish

connections between incremental proximal gradient methods and ResNets. VNs can be used to learn unrolled gradient schemes, incremental proximal schemes or TNRD methods [55]. We show numerical results of all different schemes for image denoising and deblurring, and extend VNs to 3-dimensional low-dose CT. Finally, in Chapter 6 we advocate a mean-field optimal control problem of early stopped gradient flows to determine the parameters of learnable regularizers such as the FoE and the TDV regularizer. We prove the existence of solutions of the mean-field optimal control problem in the time-continuous and time-discrete setting using a semi-implicit time-discretization. In addition, we derive stability estimates to numerically quantify the robustness of the proposed approach and we demonstrate the broad applicability to various image restoration and reconstruction problems.

Theoretical Background

2 Mathematical Preliminaries

In this chapter, we review basic mathematical concepts and important theorems of functional analysis, ordinary differential equations, probability theory, and optimization that are used throughout the thesis.

2.1 Basic Functional Analysis

In this section, we recall basic results from analysis, topology, measure theory, and functional analysis. For a detailed discussion of the different topics, we refer to [26, 84–87].

Here and in the remainder of this thesis, we consider a field $\mathbb{K}$, which is either given by the set of real numbers, i.e. $\mathbb{K} = \mathbb{R}$, or by the set of complex numbers, i.e. $\mathbb{K} = \mathbb{C}$. For any $\alpha \in \mathbb{K}$, we denote its absolute value by

$$|\alpha| \coloneqq \sqrt{\alpha\overline{\alpha}} \quad \text{with} \quad \overline{\alpha} \coloneqq \begin{cases} \mathfrak{Re}(\alpha) - \mathfrak{Im}(\alpha) & \text{if } \mathbb{K} = \mathbb{C} \\ \alpha & \text{if } \mathbb{K} = \mathbb{R} \end{cases}.$$

2.1.1 Vector Spaces

In general, a vector space is a collection of vectors that defines the addition of pairs of vectors and the scalar multiplication of a vector and a scalar. The following definition provides a detailed explanation of the properties of vector spaces.

Definition 2.1.1 (Vector Space) *A* vector space $\mathbb{V}$ *over a field* $\mathbb{K}$ *is a set of vectors such that the following holds:*

1. *Let* $x, y \in \mathbb{V}$. *We denote by* $+ : \mathbb{V} \times \mathbb{V} \to \mathbb{V}, (x, y) \mapsto x + y$ *the summation of two vectors, satisfying the following properties:*
   a) $x + y = y + x$ *for all* $x, y \in \mathbb{V}$.
   b) $x + (y + z) = (x + y) + z$ *for all* $x, y, z \in \mathbb{V}$.
   c) *There exists a unique vector* $0 \in \mathbb{V}$ *such that* $x + 0 = x$ *for any* $x \in \mathbb{V}$.
   d) *For every* $x \in \mathbb{V}$ *there exists a vector* $-x \in \mathbb{V}$ *such that* $x + (-x) = 0$.
2. *For any scalar* $\alpha \in \mathbb{K}$ *and any* $x \in \mathbb{V}$ *the scalar multiplication* $\mathbb{K} \times \mathbb{V} \to \mathbb{V}, (\alpha, x) \mapsto \alpha x$ *satisfies the following properties:*
   a) $\alpha(\beta x) = (\alpha\beta)x$ *for any* $\alpha, \beta \in \mathbb{K}$, $x \in \mathbb{V}$.
   b) $1x = x$ *for all* $x \in \mathbb{V}$.
3. *The combination of the summation and scalar multiplication operations are distributive, i.e.*
   a) $\alpha(x + y) = \alpha x + \alpha y$ *for any* $\alpha \in \mathbb{K}$, $x, y \in \mathbb{V}$.
   b) $(\alpha + \beta)x = \alpha x + \beta x$ *for any* $\alpha, \beta \in \mathbb{K}$, $x \in \mathbb{V}$.

For example, the set of n-dimensional column vectors with real components $\mathbb{R}^n$ is a vector space, where the summation and scalar multiplication are defined as

$$\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_n + y_n \end{pmatrix}, \quad \alpha \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} \alpha x_1 \\ \alpha x_2 \\ \vdots \\ x_n \end{pmatrix}$$

for $x = (x_1\ x_2\ \cdots\ x_n)^\top, y = (y_1\ y_2\ \cdots\ y_n)^\top \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$. Another important vector space is the space of real-valued matrices $\mathbb{R}^{m\times n}$ of size $m \times n$ with binary operations

$$A + B = \left(A_{ij} + B_{ij}\right)_{i=1,\dots,m}^{j=1,\dots,n}, \quad \alpha A = \left(\alpha A_{ij}\right)_{i=1,\dots,m}^{j=1,\dots,n}$$

for $A = (A_{ij})_{i=1,\dots,m}^{j=1,\dots,n}, (B_{ij})_{i=1,\dots,m}^{j=1,\dots,n} \in \mathbb{R}^{m\times n}$ and $\alpha \in \mathbb{R}$.

2.1.2 Inner Product

An inner product defined on a vector space $\mathbb{V}$ assigns to each pair of vectors $x, y \in \mathbb{V}$ a number $\langle x, y \rangle \in \mathbb{K}$ in the underlying field.

Definition 2.1.2 (Inner Product) *Let $\mathbb{V}$ be a vector space over the field $\mathbb{K}$. We call the map $(x, y) \mapsto \langle x, y \rangle_{\mathbb{V}}$ from $\mathbb{V} \times \mathbb{V}$ to $\mathbb{K}$* inner product *if it satisfies the properties:*

(S1) $\langle x, y \rangle_{\mathbb{V}} = \overline{\langle y, x \rangle}_{\mathbb{V}}$ *for any $x, y \in \mathbb{V}$.*
(S2) $\langle \alpha_1 x_1 + \alpha_2 x_2, y \rangle_{\mathbb{V}} = \alpha_1 \langle x_1, y \rangle_{\mathbb{V}} + \alpha_2 \langle x_2, y \rangle_{\mathbb{V}}$ *for any $\alpha_1, \alpha_2 \in \mathbb{K}$ and all $x_1, x_2, y \in \mathbb{V}$.*
(S3) $\langle x, x \rangle_{\mathbb{V}} \geq 0$ *for any $x \in \mathbb{V}$ and $\langle x, x \rangle = 0$ if and only if $x = 0$.*

A vector space $\mathbb{V}$ equipped with an inner product $\langle \cdot, \cdot \rangle_{\mathbb{V}}$ is called an *inner product space* $(\mathbb{V}, \langle \cdot, \cdot \rangle_{\mathbb{V}})$. The most frequently used inner product on $\mathbb{R}^n$ is the *dot product* defined for any $x, y \in \mathbb{R}^n$ by

$$\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i.$$

Throughout this thesis we equip the vector space $\mathbb{R}^n$ with the dot product unless otherwise stated. The standard inner product on $\mathbb{R}^{m\times n}$ is given by

$$\langle A, B \rangle = \operatorname{tr}\left(A^\top B\right) = \sum_{i=1}^{m} \sum_{j=1}^{n} A_{ij} B_{ij}$$

for $A, B \in \mathbb{R}^{m\times n}$. Note that $\mathbb{R}^{m\times n}$ is isomorphic to $\mathbb{R}^{mn}$ (denoted by $\mathbb{R}^{m\times n} \cong \mathbb{R}^{mn}$) with an isomorphism defined by a lexicographic ordering of the matrix elements into a vector.

2.1.3 Norm

A norm is a mapping from a vector space to the nonnegative real numbers that is positive definite, absolutely homogeneous, and obeys the triangle inequality.

Definition 2.1.3 (Norm) *Let $\mathbb{V}$ be a vector space over the field $\mathbb{K}$. A* norm *on $\mathbb{V}$ is defined as a function $\|\cdot\| : \mathbb{V} \to \mathbb{R}_+$ satisfying the properties:*

(N1) $\|x\| \geq 0$ *for any $x \in \mathbb{V}$ and $\|x\| = 0 \iff x = 0$ (positive homogeneous).*
(N2) $\|\alpha x\| = |\alpha| \, \|x\|$ *for any $x \in \mathbb{V}$ and $\alpha \in \mathbb{K}$ (absolutely homogeneous).*

(N3) $\|x + y\| \leq \|x\| + \|y\|$ *for all* $x, y \in \mathbb{V}$ *(triangle inequality).*

The pair $(\mathbb{V}, \|\cdot\|)$ is called a *normed vector space*. The finite-dimensional real vector space $\mathbb{R}^n$ equipped with an inner product $\langle \cdot, \cdot \rangle$ is called a Euclidean space if it is endowed with the norm $\|x\| = \sqrt{\langle x, x \rangle}$ induced by the inner product. This induced norm is also called the Euclidean norm.

A function $\|\cdot\| : \mathbb{V} \to \mathbb{R}$ is called a seminorm if it fulfills (N2), (N3), and (N1) without the property $(x = 0 \implies \|x\| = 0)$.

For a given center vector $y \in \mathbb{V}$ and a radius $r > 0$, we denote the open ball as

$$\mathcal{B}_{\|\cdot\|}(y, r) = \{x \in \mathbb{V} : \|x - y\| < r\}$$

and the closed ball

$$\overline{\mathcal{B}}_{\|\cdot\|}(y, r) = \{x \in \mathbb{V} : \|x - y\| \leq r\}$$

for a norm $\|\cdot\|$.

Next, we present some important norms on $\mathbb{K}^n$ and $\mathbb{K}^{m \times n}$.

For $1 \leq p \leq \infty$ the *ℓ^p-norm* on $\mathbb{K}^n$ is defined as

$$\|x\|_p := \begin{cases} \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}} & \text{if } 1 \leq p < \infty \\ \max_{i=1,\ldots,n} |x_i| & \text{if } p = \infty \end{cases}$$

for $x = (x_1\ x_2\ \cdots\ x_n)^\top \in \mathbb{K}^n$. Figure 2.1 visualizes the unit spheres $\mathcal{S}_{\|\cdot\|} := \{x \in \mathbb{R}^n : \|x\| = 1\}$ for different ℓ^p-norms in $\mathbb{R}^2$. For $p = 1$ the unit sphere is shaped like a rhombus while for $p \to \infty$ it is a square. The values in between can be used to interpolate between these two shapes. For $p = 2$ the ℓ^p-norm is equal to the usual Euclidean norm.

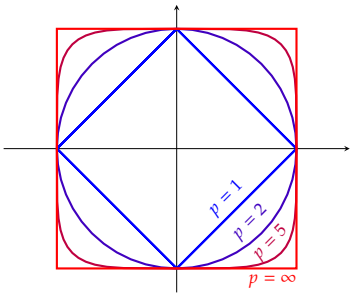


Figure 2.1: Unit spheres of ℓ^p-norms in $\mathbb{R}^2$.

An important family of matrix norms called *induced matrix norms* is based on the ℓ^p-norm of vector spaces. Let $\|\cdot\|_a$ and $\|\cdot\|_b$ be two vector norms on $\mathbb{K}^n$ and $\mathbb{K}^m$. For a matrix $A \in \mathbb{K}^{m \times n}$ the induced matrix norm is defined as

$$\|A\|_{a,b} := \max_{x \in \mathbb{K}^n : \|x\|_a \leq 1} \|Ax\|_b = \sup_{x \in \mathbb{K}^n \setminus \{0\}} \frac{\|Ax\|_b}{\|x\|_a}.$$

This definition immediately implies that for any $x \in \mathbb{K}^n$ the inequality

$$\|Ax\|_b \leq \|A\|_{a,b} \, \|x\|_a$$

holds. The choice $\|\cdot\|_a = \|\cdot\|_b = \|\cdot\|_2$ defines the *spectral norm*

$$\|A\|_2 = \|A\|_{2,2} = \sqrt{\lambda_{\max}(A^\top A)} = \sigma_{\max}(A),$$

where $A \in \mathbb{K}^{m \times n}$, $\lambda_{\max}(A^\top A)$ denotes the maximal eigenvalue of the matrix $A^\top A$ and $\sigma_{\max}(A)$ the maximal singular value of the matrix A. If $\|\cdot\|_a = \|\cdot\|_b = \|\cdot\|_1$ the induced matrix norm of a matrix $A \in \mathbb{K}^{m \times n}$ is given by

$$\|A\|_1 = \|A\|_{1,1} = \max_{j=1,\ldots,n} \sum_{i=1}^{m} |A_{ij}|,$$

which amounts to the maximal absolute column sum. Likewise, if $\|\cdot\|_a = \|\cdot\|_b = \|\cdot\|_\infty$, the induced matrix norm of a matrix $A \in \mathbb{K}^{m \times n}$ is given by

$$\|A\|_\infty = \max_{i=1,\ldots,m} \sum_{j=1}^{n} |A_{ij}|,$$

which is the maximal absolute row sum.

Another class of matrix norms accounts for the individual matrix entries. The *Frobenius norm* of a matrix $A \in \mathbb{K}^{m\times n}$ is defined as the ℓ^2-norm of all matrix elements, i.e.

$$\|A\|_F := \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|A_{ij}|^2}.$$

The $\ell^{2,1}$-norm of a matrix $A \in \mathbb{K}^{m\times n}$ is defined as

$$\|A\|_{2,1} := \sum_{j=1}^{n}\sqrt{\sum_{i=1}^{m}|A_{ij}|^2}.$$

It is essentially the sum of the ℓ^2-norms of the columns of a matrix. Later in this thesis, we will use this norm to define the isotropic discrete total variation (TV).

2.1.4 Basic Topology

In this section, we briefly recall the basic concepts of topology and define the notation used throughout this thesis.

Let $(\mathbb{V}, \|\cdot\|)$ be a normed vector space. Then, a set $\mathcal{S} \subset \mathbb{V}$ is

- *bounded* if there exists a radius $r > 0$ such that $\mathcal{S} \subset \mathcal{B}_{\|\cdot\|}(0, r)$.
- *open* if for all $x \in \mathcal{S}$ there exists $\varepsilon > 0$ such that $\mathcal{B}_{\|\cdot\|}(x, \varepsilon) \subset \mathcal{S}$.
- *closed* if $\mathbb{V} \setminus \mathcal{S}$ is open.
- *compact* if every open cover of $\mathcal{S}$ has a finite subcover, i.e. for every family of open sets $\mathcal{S}_i \subset \mathbb{V}$, $i \in \mathbb{N}$, with $\mathcal{S} \subset \bigcup_{i\in\mathbb{N}} \mathcal{S}_i$, there are finitely many $\mathcal{S}_{i_1}, \ldots, \mathcal{S}_{i_N}$ such that $\mathcal{S} \subset \bigcup_{j=1}^{N} \mathcal{S}_{i_j}$.

Theorem 2.1.1 *Let $\mathbb{V} = \mathbb{K}^n$ and $\mathcal{S} \subset \mathbb{V}$. The following statements are equivalent:*

- *$\mathcal{S}$ is compact.*
- *$\mathcal{S}$ is bounded and closed.*
- *Every sequence in $\mathcal{S}$ has a convergent subsequence whose limit is in $\mathcal{S}$.*

Proof. See [85, Section 4.6, 4.7, Heine-Borel Theorem]. □

Let $(\mathbb{V}, \|\cdot\|)$ be a normed vector space and $\mathcal{S} \subset \mathbb{V}$.

- A point $x \in \mathcal{S}$ is an *interior point* if there exists an open set $\mathcal{U} \subset \mathcal{S}$ such that $x \in \mathcal{U}$. The *interior* of $\mathcal{S}$ (denoted by $\operatorname{int}(\mathcal{S})$) is the set of all interior points of $\mathcal{S}$.
- A point $x \in \mathbb{V}$ is a *point of closure* of $\mathcal{S}$ if for every open set $\mathcal{U} \subset \mathbb{V}$ with $x \in \mathcal{U}$ there exists $s \in \mathcal{S}$ such that $s \in \mathcal{U}$. The *closure* of $\mathcal{S}$ (denoted by $\overline{S}$) is the set of all points of closure of $\mathcal{S}$.
- The *boundary* of $\mathcal{S}$ is the set $\partial\mathcal{S} := \overline{\mathcal{S}} \setminus \operatorname{int}(\mathcal{S})$.

2.1.5 Convergence and Continuity

Here, we define the basic concept of convergence of sequences and relate it to the notion of completeness of a space. Then we show how the continuity of a function relates a converging sequence of its argument to a converging sequence of the corresponding images.

Let $(\mathbb{V}, \|\cdot\|)$ be a normed vector space. A *sequence* is a map $x : \mathbb{N} \to \mathbb{V}$, which we denote by the shorthand $x^n := x(n)$.

Definition 2.1.4 (Convergence) *A point $x_0 \in \mathbb{V}$ is the* limit *of a sequence x^n if for all $\varepsilon > 0$ there exists an $N \in \mathbb{N}$ such that for every $i \geq N$ the inequality $\left\|x^i - x_0\right\| < \varepsilon$ holds true. Then, we say that the sequence x^n converges to x_0 and write $x^n \to x_0$ as $n \to \infty$ or*

$$\lim_{n\to\infty} x^n = x_0.$$

To relate the completeness of a space to converging sequences, we characterize Cauchy sequences in the next definition.

Definition 2.1.5 (Cauchy Sequence) *The sequence x^n is a* Cauchy sequence *if for every $\varepsilon > 0$ there exists an $N \in \mathbb{N}$ such that for all $i, j \geq N$*

$$\left\|x^i - x^j\right\| < \varepsilon$$

holds true.

Thus, a Cauchy sequence is a sequence in which the distance in terms of the norm $\|\cdot\|$ between its elements becomes arbitrarily small as the sequence progresses. If x^n is a sequence in $\mathbb{V}$, then a point $x \in \mathbb{V}$ is called a *cluster point* of x^n if there exists a *subsequence* x^{n^i}, i.e. a sequence n^i in $\mathbb{N}$ with $n^i \to \infty$ as $i \to \infty$, such that $x = \lim_{i\to\infty} x^{n^i}$. Consequently, a Cauchy sequence can have at most a single cluster point and thereby has at most a single converging subsequence.

Definition 2.1.6 (Completeness) *The space $(\mathbb{V}, \|\cdot\|)$ is* complete *if every Cauchy sequence converges to an element of $\mathbb{V}$.*

A *Hilbert space* is an inner product space which is complete w.r.t. the induced norm $\|\cdot\| : \mathbb{V} \to \mathbb{R}_+, x \mapsto \sqrt{\langle x, x\rangle}$. In analogy, a *Banach space* is a normed space that is complete w.r.t. its norm.

Let $(\mathbb{V}, \|\cdot\|_{\mathbb{V}})$ and $(\mathbb{W}, \|\cdot\|_{\mathbb{W}})$ be two normed vector spaces. To state the next theorem, we first have to define continuous functions.

Definition 2.1.7 (Continuity) *A function $f : \mathbb{V} \to \mathbb{W}$ is* continuous *at a point $x_0 \in \mathbb{V}$ if for all $\varepsilon > 0$ there exists a $\delta > 0$ such that $\|x - x_0\|_{\mathbb{V}} < \delta$ implies $\left\|f(x) - f(x_0)\right\|_{\mathbb{W}} < \varepsilon$. The function f is continuous on $\mathbb{V}$ if f is continuous for all $x_0 \in \mathbb{V}$.*

The following theorem links the continuity of a function with the convergence of the images of a sequence.

Theorem 2.1.2 *Let $f : \mathbb{V} \to \mathbb{W}$ be a function. The following statements are equivalent:*

1. *The function f is continuous.*
2. *For every set $\mathcal{S} \subseteq \mathbb{W}$, which is open in $\mathbb{W}$, the set $f^{-1}(\mathcal{S})$ is open in $\mathbb{V}$.*
3. *For every $x_0 \in \mathbb{V}$ and every sequence $x^n \in \mathbb{V}$ such that $x^n \to x_0$ in $\mathbb{V}$ the sequence $f(x^n)$ converges to $f(x_0)$ in $\mathbb{W}$ as $n \to \infty$.*

Proof. See [85, Section 2.17]. □

A generalization of the concept of continuity is given by Lipschitz continuity, which will be frequently used in this thesis.

Definition 2.1.8 (Lipschitz Continuity) *A function $f : \mathbb{V} \to \mathbb{W}$ is called* Lipschitz

continuous *with Lipschitz constant* $L > 0$ *if*

$$\left\|f(x) - f(y)\right\|_{\mathbb{W}} \leq L \left\|x - y\right\|_{\mathbb{V}} \quad \forall x, y \in \mathbb{V}.$$

2.1.6 Dual Space

To characterize the dual space of a vector space, we first need to define a linear transform.

Definition 2.1.9 (Linear Transform) *Given two vector spaces* $\mathbb{V}, \mathbb{W}$ *over a field* $\mathbb{K}$*, a function* $\mathrm{A} : \mathbb{V} \to \mathbb{W}$ *is called a* linear transform *if for all* $x, y \in \mathbb{V}$ *and* $\alpha, \beta \in \mathbb{K}$

$$\mathrm{A}(\alpha x + \beta y) = \alpha \mathrm{A}(x) + \beta \mathrm{A}(y)$$

holds.

Note that on a finite-dimensional vector space all linear transforms $\mathrm{A} : \mathbb{K}^n \to \mathbb{K}^m$ are uniquely described by a matrix $A \in \mathbb{K}^{m \times n}$, i.e. $\mathrm{A}(x) = Ax$ for all $x \in \mathbb{K}^n$.

Definition 2.1.10 (Dual Space) *A* linear functional *on a vector space* $\mathbb{V}$ *is a linear transformation that maps from* $\mathbb{V}$ *to* $\mathbb{K}$*. The set of all linear functionals on* $\mathbb{V}$ *defines the* dual space *denoted by* $\mathbb{V}^*$*.*

Along with the dual space, we define the dual norm by means of the duality pairing.

Definition 2.1.11 (Dual Norm) *Let* $\|\cdot\|$ *be a norm on the vector space* $\mathbb{V}$*. The corresponding* dual norm $\|\cdot\|_* : \mathbb{V}^* \to \mathbb{R}_+$ *is defined as*

$$\left\|y\right\|_* \coloneqq \max_{x \in \mathbb{V} : \|x\| \leq 1} \left\langle x, y \right\rangle,$$

where $\left\langle x, y \right\rangle \coloneqq y(x)$ *denotes the* duality pairing*.*

On a finite-dimensional vector space $\mathbb{K}^n$ the dual norm of the ℓ^p-norm is given by the ℓ^q-norm with $\frac{1}{p} + \frac{1}{q} = 1$ for $p, q \in [1, \infty]$ using the convention $\frac{1}{\infty} = 0$. Thus, the ℓ^2-norm is self-dual ($p = q = 2$) and the dual norm of the ℓ^1-norm is the ℓ^∞-norm ($p = 1, q = \infty$) and vice versa.

A fundamental representation theorem of functional analysis that links the dual space with its associated vector space is due to Riesz.

Theorem 2.1.3 (Riesz' Representation) *Let* $\mathbb{V}$ *be a Hilbert space. Then, for any linear functional* $y \in \mathbb{V}^*$ *there exists a unique* $v \in \mathbb{V}$ *such that*

$$y(x) = \langle x, v \rangle_{\mathbb{V}} \quad \forall x \in \mathbb{V},$$

where $\langle \cdot, \cdot \rangle_{\mathbb{V}}$ *denotes the inner product on* $\mathbb{V}$*. In particular, we have* $\left\|y\right\|_* = \|v\|$ *and* $\mathbb{V}^*$ *is isomorphic to* $\mathbb{V}$ *(*$\mathbb{V}^* \cong \mathbb{V}$*).*

Proof. See [87, Section 7.3, Theorem 7.26]. □

Due to this relation, the dual space $\mathbb{V}^*$ of a vector space $\mathbb{V}$ is also a vector space. The dual space of the dual space is the so-called *bidual space* $\mathbb{V}^{**}$, which is equal to the original space for finite-dimensional vector spaces. Moreover, it follows that the *bidual norm* $\|\cdot\|_{**}$ is the same as the original norm $\|\cdot\|$ in the finite-dimensional setting.

Finally, we define the adjoint transform of a linear transformation on Hilbert spaces.

Definition 2.1.12 (Adjoint Transform) *Let* $(\mathbb{V}, \langle \cdot, \cdot \rangle_{\mathbb{V}})$ *and* $(\mathbb{W}, \langle \cdot, \cdot \rangle_{\mathbb{W}})$ *be two Hilbert spaces and* $\mathrm{A} : \mathbb{V} \to \mathbb{W}$ *a linear transform. Then the* adjoint linear transform $\mathrm{A}^* : \mathbb{W}^* \to \mathbb{V}^*$ *is defined via*

$$\left\langle \mathrm{A}(x), y \right\rangle_{\mathbb{W}} = \left\langle x, \mathrm{A}^*(y) \right\rangle_{\mathbb{V}^*}.$$

Let $\mathbb{V}, \mathbb{W}$ be Hilbert spaces. Then the adjoint linear transform has the properties:

1. $(\alpha \mathrm{A}_1(x) + \mathrm{A}_2(x))^* = \overline{\alpha} \mathrm{A}_1^*(x) + \mathrm{A}_2^*(x)$ for all $\alpha \in \mathbb{K}$ and linear transforms $\mathrm{A}_1, \mathrm{A}_2 : \mathbb{V} \to \mathbb{W}$.
2. $\mathrm{A}^{**} = \mathrm{A}$ for any linear transform $\mathrm{A} : \mathbb{V} \to \mathbb{W}$.
3. $(\mathrm{A}_1 \mathrm{A}_2)^* = \mathrm{A}_2^* \mathrm{A}_1^*$ for all linear transforms $\mathrm{A}_1, \mathrm{A}_2 : \mathbb{V} \to \mathbb{V}$.

As an example, let us consider $\mathbb{V} = \mathbb{K}^n$ and $\mathbb{W} = \mathbb{K}^m$ both equipped with the dot product. Hence, every linear transform $\mathrm{A} : \mathbb{K}^n \to \mathbb{K}^m$ is represented by a corresponding matrix $A \in \mathbb{K}^{m \times n}$ such that $\mathrm{A}(x) = Ax$. Then, the adjoint transform is given by

$$\left\langle Ax, y \right\rangle = \left\langle x, A^* y \right\rangle,$$

where $A^* = A^\top$ if $\mathbb{K} = \mathbb{R}$ or $A^* = \overline{A^\top}$ if $\mathbb{K} = \mathbb{C}$. In this thesis, we frequently denote a linear transform as a linear operator and its adjoint linear transform as the corresponding adjoint operator.

2.1.7 Extended Real-valued Functions and Closedness

In this section, we define *extended real-valued functions* as functions over the entire underlying space that can attain any real value including $-\infty$ and ∞. Thus, their range is $\overline{\mathbb{R}} := \mathbb{R} \cup \{-\infty, \infty\}$. For this general class of functions, we define the notion of closedness and relate it to the continuity of the function. We conclude this section by presenting two theorems that ensure the existence of minimizers of closed extended real-valued functions.

Let $\mathbb{V}$ be a Banach space over a field $\mathbb{K}$.

Definition 2.1.13 (Proper) *An extended real-valued function* $f : \mathbb{V} \to \overline{\mathbb{R}}$ *is* proper *if* f *never takes the value* $-\infty$ *and is not identical to* ∞.

Since we are mostly interested in computing minimizers of extended real-valued functions, almost all functions of interest are proper. Next, we define the (effective) domain of an extended real-valued function as the set of arguments that are not mapped to ∞.

Definition 2.1.14 (Domain) *The domain of a proper extended real-valued function* $f : \mathbb{V} \to \overline{\mathbb{R}}$ *is defined as*

$$\operatorname{dom}(f) := \{x \in \mathbb{V} : f(x) < \infty\}.$$

One of the simplest non-trivial examples of extended real-valued functions are indicator functions. For a given subset $\mathcal{S} \subset \mathbb{V}$, we denote by $\delta_{\mathcal{S}} : \mathbb{V} \to (-\infty, \infty]$ its *indicator function*, which is defined as

$$\delta_{\mathcal{S}}(x) := \begin{cases} 0 & \text{if } x \in \mathcal{S} \\ \infty & \text{if } x \notin \mathcal{S} \end{cases}. \tag{2.1}$$

Clearly, the domain of an indicator function is given by $\operatorname{dom}(\delta_{\mathcal{S}}) = \mathcal{S}$.

To link the notion of closedness of functions with its equivalent for sets, we define the epigraph of an extended real-valued function.

Definition 2.1.15 (Epigraph) *The* epigraph *of an extended real-valued function* $f : \mathbb{V} \to \overline{\mathbb{R}}$ *is defined as*

$$\operatorname{epi}(f) := \{(x, y) \in \mathbb{V} \times \mathbb{R} : f(x) \leq y\}.$$

Figure 2.2 illustrates the concept of an epigraph. Note that if $(x, y) \in \operatorname{epi}(f)$, then $x \in \operatorname{dom}(f)$.

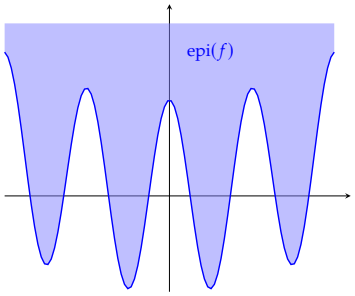


Figure 2.2: Visualization of the epigraph of $f(x) = \frac{1}{2}x^2 + \cos(4\pi x)$.

Definition 2.1.16 (Closedness) *A function* $f : \mathbb{V} \to \overline{\mathbb{R}}$ *is* closed *if its epigraph is closed.*

A property of a function that is equivalent to closedness is lower semicontinuity as we will see in the subsequent theorem.

Definition 2.1.17 (Lower Semicontinuity) *A function* $f : \mathbb{V} \to \overline{\mathbb{R}}$ *is* lower semicontinuous *at* $x \in \mathbb{V}$ *if*

$$f(x) \leq \liminf_{n \to \infty} f(x^n)$$

for any sequence $x^n \in \mathbb{V}$ *for which* $x^n \to x$ *as* $n \to \infty$. *A function* $f : \mathbb{V} \to \overline{\mathbb{R}}$ *is* lower semicontinuous *if it is lower semicontinuous at every point in* $\mathbb{V}$.

In this definition, we use the limes inferior which is the smallest cluster point of a sequence and defines the largest lower bound. For a sequence x^n, the limes inferior is formally defined as

$$\liminf_{n \to \infty} x^n := \sup_{n \in \mathbb{N}} \inf_{k \geq n} x_k.$$

Theorem 2.1.4 (Closedness) *Let* $\mathbb{V}$ *be a normed vector space and* $f : \mathbb{V} \to \overline{\mathbb{R}}$. *Then the following claims are equivalent:*

1. f *is closed.*
2. f *is lower semicontinuous.*
3. *For any* $\alpha \in \mathbb{R}$, *the level set defined as*

$$\operatorname{Lev}(f, \alpha) := \left\{x \in \mathbb{V} : f(x) \leq \alpha\right\}$$

is closed.

Proof. See [84, Section 2.1, Theorem 2.6]. □

In the next theorem, we establish a relationship between continuous functions defined over a closed domain and the property of closedness.

Theorem 2.1.5 (Continuity implies Closedness) *Let* $f : \mathbb{V} \to (-\infty, \infty]$ *be an extended real-valued function that is continuous over its domain and suppose that* $\operatorname{dom}(f)$ *is closed. Then,* f *is closed.*

Proof. See [84, Section 2.2, Theorem 2.8]. □

Note that closedness of a function does not require that it is continuous, cf. indicator functions.

Based on the previous definitions, we can state Weierstrass' theorem that ensures the existence of minimizers of a closed function over a compact set.

Theorem 2.1.6 (Weierstrass' Theorem for Closed Functions) *Let $f : \mathbb{V} \to (-\infty, \infty]$ be a proper closed function and assume that $\mathcal{S}$ is a compact set satisfying $\operatorname{dom}(f) \cap \mathcal{S} \neq \emptyset$. Then*

1. *f is bounded from below over $\mathcal{S}$,*
2. *f attains its minimal value over $\mathcal{S}$.*

Proof. See [84, Section 2.2, Theorem 2.12]. □

If $\mathcal{S}$ is not compact, then Weierstrass' theorem does not hold. However, we can replace the compactness of $\mathcal{S}$ by the *coerciveness* of f to show the attainment of a minimizer over an arbitrary closed set.

Definition 2.1.18 (Coerciveness) *A proper function $f : \mathbb{V} \to (-\infty, \infty]$ is called* coercive *if*

$$\lim_{\|x\| \to \infty} f(x) = \infty.$$

Theorem 2.1.7 (Attainment under Coerciveness) *Let $f : \mathbb{V} \to (-\infty, \infty]$ be a closed, proper and coercive function and let $\mathcal{S} \subseteq \mathbb{V}$ be a non-empty closed set satisfying $\mathcal{S} \cap \operatorname{dom}(f) \neq \emptyset$. Then f attains its minimal value over $\mathcal{S}$.*

Proof. See [84, Section 2.2, Theorem 2.14]. □

2.1.8 Space of Continuous and Differentiable Functions

In this section, we introduce the infinite vector space of continuous functions as well as the space of k-times continuously differentiable functions.

In the following, let $\mathcal{S} \subset \mathbb{K}^n$ be an open and bounded set and $(\mathcal{S}', \|\cdot\|)$ a Banach space for $\mathcal{S}' \subset \mathbb{K}^m$.

Definition 2.1.19 (Space of Continuous Functions) *We denote by $C^0(\overline{\mathcal{S}}, \mathcal{S}')$ the set of* continuous functions *mapping from $\overline{\mathcal{S}}$ to $\mathcal{S}'$. Then, $C^0(\overline{\mathcal{S}}, \mathcal{S}')$ equipped with the supremum norm*

$$\left\|f\right\|_{C^0(\overline{\mathcal{S}}, \mathcal{S}')} := \sup_{x \in \overline{\mathcal{S}}} \left\|f(x)\right\|$$

becomes a Banach space.

To define the space of continuously differentiable functions, we first need to define the partial derivative w.r.t. a multi-index. A *multi-index* $\alpha = (\alpha_1 \ \cdots \ \alpha_n)^\top \in \mathbb{N}_0^n$ of order k is an n-dimensional vector of non-negative integers such that

$$k = |\alpha| := \sum_{i=1}^{n} \alpha_i.$$

For a multi-index $\alpha \in \mathbb{N}_0^n$ we define the *partial derivative* of $f : \mathcal{S} \to \mathcal{S}'$ at $x \in \mathcal{S}$ as

$$\partial^\alpha f(x) \coloneqq \frac{\partial^{|\alpha|} f}{\partial x_1^{\alpha_1} \cdots \partial x_n^{\alpha_n}}(x).$$

For $l \in \mathbb{N}_0$, we define the number of possible multi-indices of order l and up to order l as

$$\mathrm{N}(l) \coloneqq |\{\alpha \in \mathbb{N}_0^n : |\alpha| = l\}|, \quad \overline{\mathrm{N}}(l) \coloneqq |\{\alpha \in \mathbb{N}_0^n : |\alpha| \leq l\}|,$$

respectively. To sort the set of multi-indices $\mathbb{N}_0^n$, we use a total ordering defined in the following manner: $\alpha < \beta$ if either $|\alpha| < |\beta|$, or if $|\alpha| = |\beta|$, then there exists an index $1 \leq k \leq n$ such that $\alpha_i = \beta_i$ for $1 \leq i < k$ and $\alpha_k < \beta_k$.

Using the concept of multi-indices, we define the l^{th}-order *gradient* of a k-times differentiable function $f : \mathcal{S} \to \mathcal{S}'$ as the vector-valued function

$$D^l f \coloneqq (\partial^\alpha f)_{|\alpha|=l} : \mathcal{S} \to \mathbb{K}^{\mathrm{N}(l) \times m},$$

where the j^{th} component of $D^l f$ is the j^{th} partial derivative of order l of f w.r.t. the ordering on $\mathbb{N}_0^n$ defined above.

Definition 2.1.20 (Space of Differentiable Functions) *For an order $k \in \mathbb{N}$, we denote by $C^k(\mathcal{S}, \mathcal{S}')$ the set of all k-times* continuously differentiable functions *mapping from $\mathcal{S}$ to $\mathcal{S}'$. Then, $C^k(\mathcal{S}, \mathcal{S}')$ equipped with the norm*

$$\|f\|_{C^k(\mathcal{S},\mathcal{S}')} \coloneqq \sum_{|\alpha| \leq k} \|\partial^\alpha f\|_{C^0(\mathcal{S},\mathcal{S}')}$$

becomes a Banach space.

Throughout this thesis, we use the shorthand $C^k(\overline{\mathcal{S}}, \mathcal{S}') = C^0(\overline{\mathcal{S}}, \mathcal{S}') \cap C^k(\mathcal{S}, \mathcal{S}')$. The vector space of *infinitely differentiable functions* from $\mathcal{S}$ to $\mathcal{S}'$ is defined as

$$C^\infty(\mathcal{S}, \mathcal{S}') \coloneqq \bigcap_{k \in \mathbb{N}} C^k(\mathcal{S}, \mathcal{S}').$$

Another class of continuous functions that are frequently required for solving dynamical systems are given by Hölder function spaces.

Definition 2.1.21 (Hölder Spaces) *We define the* Hölder space *of order $k \in \mathbb{N}$ and exponent $s \in (0, 1]$ as*

$$C^{k,s}(\overline{\mathcal{S}}, \mathcal{S}') \coloneqq \{f \in C^k(\overline{\mathcal{S}}, \mathcal{S}') : \|f\|_{C^{k,s}(\overline{\mathcal{S}},\mathcal{S}')} < \infty\}, \tag{2.2}$$

where the Hölder norm reads as

$$\|f\|_{C^{k,s}(\overline{\mathcal{S}},\mathcal{S}')} \coloneqq \sum_{|\alpha| \leq k} \|\partial^\alpha f\|_{C^0(\overline{\mathcal{S}},\mathcal{S}')} + \sum_{|\alpha|=k} \sup_{\substack{x,y \in \overline{\mathcal{S}} \\ x \neq y}} \frac{\|\partial^\alpha f(x) - \partial^\alpha f(y)\|}{\|x - y\|^s}. \tag{2.3}$$

Every Hölder space is a Banach space.

In contrast to the space of k-times continuously differentiable functions, the Hölder space additionally ensures that the k^{th} derivative is Hölder continuous with parameter s. In particular, if $s = 1$, the Hölder continuity is equivalent to Lipschitz continuity.

2.1.9 Measure Theory

In this section, we briefly recall relevant concepts and properties from measure theory in order to develop the space of Lebesgue integrable functions and lay the basis for the probability theory in the next section.

In the following, we consider an arbitrary non-empty set $\mathcal{S}$ and denote by $2^{\mathcal{S}} = \{\mathcal{A} : \mathcal{A} \subset \mathcal{S}\}$ its *power set* consisting of all subsets of $\mathcal{S}$.

Definition 2.1.22 (Classes of Sets) *A class of sets $\mathfrak{A} \subset 2^{\mathcal{S}}$ is called*

- *$\cap$-closed (closed under intersections) if $\mathcal{A} \cap \mathcal{B} \in \mathfrak{A}$ for any $\mathcal{A}, \mathcal{B} \in \mathfrak{A}$.*
- *σ-$\cap$-closed (closed under countable intersections) if $\bigcap_{i\in\mathbb{N}} \mathcal{A}_i \in \mathfrak{A}$ for any choice of countably many sets $\mathcal{A}_1, \mathcal{A}_2, \ldots \in \mathfrak{A}$.*
- *$\cup$-closed (closed under unions) if $\mathcal{A} \cup \mathcal{B} \in \mathfrak{A}$ for any $\mathcal{A}, \mathcal{B} \in \mathfrak{A}$.*
- *σ-$\cup$-closed (closed under countable unions) if $\bigcup_{i\in\mathbb{N}} \mathcal{A}_i \in \mathfrak{A}$ for any choice of countably many sets $\mathcal{A}_1, \mathcal{A}_2, \ldots \in \mathfrak{A}$.*
- *$\setminus$-closed (closed under differences) if $\mathcal{A} \setminus \mathcal{B} \in \mathfrak{A}$ for any $\mathcal{A}, \mathcal{B} \in \mathfrak{A}$.*
- *closed under complements if $\mathcal{A}^C := \mathcal{S} \setminus \mathcal{A} \in \mathfrak{A}$ for any $\mathcal{A} \in \mathfrak{A}$.*

Here, the term 'countable' means either finite or countably infinite.

A particularly interesting class of sets is given by the σ-algebra, which will be later used to define observable random events.

Definition 2.1.23 (σ-algebra) *A class of sets $\mathfrak{A} \subset 2^{\mathcal{S}}$ is called a* σ-algebra *if it fulfills the following conditions:*

- *$\mathcal{S} \in \mathfrak{A}$*
- *$\mathfrak{A}$ is closed under complements.*
- *$\mathfrak{A}$ is closed under countable unions.*

The next theorem states that for a given class of sets one can generate a corresponding σ-algebra.

Theorem 2.1.8 (Generated σ-algebra) *Let $\mathfrak{E} \subset 2^{\mathcal{S}}$. Then there exists a smallest σ-algebra $\sigma(\mathfrak{E})$ such that*

$$\sigma(\mathfrak{E}) := \bigcap_{\mathfrak{A}\subset 2^{\mathcal{S}}:\mathfrak{E}\subset\mathfrak{A},\mathfrak{A} \text{ is a } \sigma\text{-algebra}} \mathfrak{A}.$$

$\sigma(\mathfrak{E})$ is called the σ-algebra generated *by $\mathfrak{E}$.*

Proof. See [87, Section 1.1, Theorem 1.16]. □

The pair $(\mathcal{S}, \mathfrak{A})$ consisting of a non-empty set $\mathcal{S}$ and a σ-algebra $\mathfrak{A} \subset 2^{\mathcal{S}}$ is called a *measurable space* and the sets $\mathcal{A} \in \mathfrak{A}$ are called *measurable sets*. Let $(\mathcal{S}, \|\cdot\|)$ be a metric space. The σ-algebra $\mathfrak{B}(\mathcal{S})$ that is generated by all open sets is called the *Borel σ-algebra* on $\mathcal{S}$ and its elements $\mathcal{A} \in \mathfrak{B}(\mathcal{S})$ are called *Borel measurable sets*.

To quantify an element of a class of sets, we introduce set functions.

Definition 2.1.24 (Set Functions) *Let $\mathfrak{A} \subset 2^{\mathcal{S}}$ and let $\mu : \mathfrak{A} \to [0, \infty]$ be a* set function. *We say that μ is*

- monotone *if $\mu(\mathcal{A}) \leq \mu(\mathcal{B})$ for any $\mathcal{A}, \mathcal{B} \in \mathfrak{A}$ with $\mathcal{A} \subset \mathcal{B}$.*
- additive *if $\mu(\bigcup_{i=1}^n \mathcal{A}_i) = \sum_{i=1}^n \mu(\mathcal{A}_i)$ for any mutually disjoint $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n \in \mathfrak{A}$ with $\bigcup_{i=1}^n \mathcal{A}_i \in \mathfrak{A}$.*

- σ-additive *if* $\mu(\bigcup_{i\in\mathbb{N}} \mathcal{A}_i) = \sum_{i\in\mathbb{N}} \mu(\mathcal{A}_i)$ *for any choice of countably many mutually disjoint* $\mathcal{A}_1, \mathcal{A}_2, \ldots \in \mathfrak{A}$ *with* $\bigcup_{i\in\mathbb{N}} \mathcal{A}_i \in \mathfrak{A}$.
- subadditive *if for any finite choice of sets* $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n \in \mathfrak{A}$ *with* $\bigcup_{i=1}^n \mathcal{A}_i \in \mathfrak{A}$, *we have* $\mu(\bigcup_{i=1}^n \mathcal{A}_i) \leq \sum_{i=1}^n \mu(\mathcal{A}_i)$.
- σ-subadditive *if for any countable choice of sets* $\mathcal{A}_1, \mathcal{A}_2, \ldots \in \mathfrak{A}$ *with* $\bigcup_{i\in\mathbb{N}} \mathcal{A}_i \in \mathfrak{A}$, *we have* $\mu(\bigcup_{i\in\mathbb{N}} \mathcal{A}_i) \leq \sum_{i\in\mathbb{N}} \mu(\mathcal{A}_i)$.

Definition 2.1.25 (Measure) *Let* $\mathfrak{A} \subset 2^{\mathcal{S}}$ *be a* σ*-algebra. We call a function* $\mu : \mathfrak{A} \to [0, \infty]$ *a* measure *on* $\mathfrak{A}$ *if* $\mu(\emptyset) = 0$ *and* μ *is* σ*-subadditive.*

A measure is called σ-finite *if there exists a sequence of sets* $\mathcal{S}^n \in \mathfrak{A}$ *with* $\mu(\mathcal{S}^n) < \infty$ *for* $n \in \mathbb{N}$ *such that* $\mathcal{S} = \bigcup_{n\in\mathbb{N}} \mathcal{S}^n$.

As an example let $\mathcal{S} = \mathbb{R}^n$, we denote the semiring of half open cuboids by

$$\mathfrak{A} = \{(a, b] : a, b \in \mathbb{R}^n, a < b\},$$

which are sets of the form

$$(a, b] \coloneqq \{x \in \mathbb{R}^n : a_i < x_i \leq b_i \text{ for } i = 1, \ldots, n\}.$$

Here, for $a, b \in \mathbb{R}^n$, we write $a < b$ if $a_i < b_i$ for $i = 1, \ldots n$. Then the measure accounting for the volume of a cuboid is defined as

$$\mu((a, b]) = \mu\left(\bigtimes_{i=1}^n (a_i, b_i]\right) \coloneqq \prod_{i=1}^n (b_i - a_i).$$

with the value being ∞ if $b_i = \infty$ or $a_i = -\infty$ for any $i = 1, \ldots, n$. The extension theorem of measures [87, Section 1.3, Theorem 1.53] yields the well-known Lebesgue measure:

Theorem 2.1.9 (Lebesgue Measure) *There exists a uniquely determined measure* L^n *on* $(\mathbb{R}^n, \mathfrak{B}(\mathbb{R}^n))$ *with the property that*

$$\mathrm{L}^n((a, b]) = \prod_{i=1}^n (b_i - a_i)$$

for all $a, b \in \mathbb{R}^n$ *with* $a < b$. L^n *is called the* Lebesgue measure *on* $(\mathbb{R}^n, \mathfrak{B}(\mathbb{R}^n))$.

The combination of a measurable space with a measure results in a measure space, as the following definition states.

Definition 2.1.26 (Measure Space) *Let* $\mathcal{S}$ *be an arbitrary set,* $\mathfrak{A} \subset 2^{\mathcal{S}}$ *a* σ*-algebra and* μ *a measure on* $\mathfrak{A}$. *We denote by* $(\mathcal{S}, \mathfrak{A}, \mu)$ *the corresponding* measure space.

Next, we define the completeness property of a measure space by means of μ-null sets.

Definition 2.1.27 (μ-null Set and Completeness) *Let* $(\mathcal{S}, \mathfrak{A}, \mu)$ *be a measure space. A set* $N \in \mathfrak{A}$ *is called a* μ-null set *or null set if* $\mu(N) = 0$. *We denote by* N_μ *the class of all subsets of* μ*-null sets. A measure space* $(\mathcal{S}, \mathfrak{A}, \mu)$ *is* complete *if* $N_\mu \subset \mathfrak{A}$.

The following remark presents a general concept to complete a measure space by including all subsets of μ-null set to its underlying σ-algebra.

Remark 2.1.1 (Completion of a Measure Space) Let $(\mathcal{S}, \mathfrak{A}, \mu)$ be a σ-finite measure space. There exists a unique smallest σ-algebra $\mathfrak{A}^* \supset \mathfrak{A}$ and an extension μ^* of μ to $\mathfrak{A}^*$ such that $(\mathcal{S}, \mathfrak{A}^*, \mu^*)$ is complete. This *completion* of $(\mathcal{S}, \mathfrak{A}, \mu)$ is given by

$$\mathfrak{A}^* = \sigma(\mathfrak{A} \cup N_\mu), \quad \mu^*(\mathcal{A} \cup N) = \mu(\mathcal{A})$$

for any $\mathcal{A} \in \mathfrak{A}$ and $N \in N_\mu$.

Hence, the resulting σ-algebra depends on the used measure.

Let $E(x)$ be a property that a point $x \in \mathcal{S}$ may have or not. We write that E holds μ-almost everywhere (a.e.) if there exists a μ-null set N such that $E(x)$ holds for all $x \in \mathcal{S} \setminus N$.

In the following, we consider a complete measure space $(\mathcal{S}, \mathfrak{A}, \mu)$ for a set $\mathcal{S} \subset \mathbb{R}^n$ and σ-algebra $\mathfrak{A} \subset 2^{\mathcal{S}}$ to introduce measurable functions and their image measure.

Definition 2.1.28 (Measurable Functions) *Let $(\mathcal{S}, \mathfrak{A})$ and $(\mathcal{S}', \mathfrak{A}')$ be measurable spaces. A function $f : \mathcal{S} \to \mathcal{S}'$ is called* measurable *if $f^{-1}(\mathfrak{A}') \coloneqq \left\{f^{-1}(\mathcal{A}') : \mathcal{A}' \in \mathfrak{A}'\right\} \subset \mathfrak{A}$, that is if*

$$f^{-1}(\mathcal{A}') \in \mathfrak{A}$$

for any $\mathcal{A}' \in \mathfrak{A}'$.

Every measurable function introduces an induced image measure according to the subsequent definition.

Definition 2.1.29 (Image Measure) *Let $(\mathcal{S}, \mathfrak{A})$ and $(\mathcal{S}', \mathfrak{A}')$ be measurable spaces and let μ be a measure on $(\mathcal{S}, \mathfrak{A})$. Further, let $f : (\mathcal{S}, \mathfrak{A}) \to (\mathcal{S}', \mathfrak{A}')$ be measurable. The* image measure *of μ under the map f is the measure $\mu \circ f^{-1}$ on $(\mathcal{S}', \mathfrak{A}')$ and is defined as*

$$\mu \circ f^{-1} : \mathfrak{A}' \to [0, \infty], \quad \mathcal{A}' \mapsto \mu(f^{-1}(\mathcal{A}')).$$

In the remainder, we only consider real numbers $\mathbb{K} = \mathbb{R}$ to develop the space of integrable functions mapping from $\mathbb{R}^n$ to $\mathbb{R}^m$. However, if one considers complex-valued functions defined over a complex domain, all integrals have to be performed entry-wise and separate for the real and imaginary parts.

To define the space of integrable functions, we first need to define the Lebesgue integral in analogy to [87, Section 4]. To this end, we start by introducing simple functions as a weighted sum of indicator functions. For an arbitrary set $\mathcal{A} \in \mathfrak{A} \subset 2^{\mathcal{S}}$, the associated indicator function $\mathbb{1}_{\mathcal{A}}(x) : \mathcal{S} \to \mathbb{R}$ is defined as

$$\mathbb{1}_{\mathcal{A}}(x) \coloneqq \begin{cases} 1 & \text{if } x \in \mathcal{A} \\ 0 & \text{if } x \notin \mathcal{A} \end{cases}. \tag{2.4}$$

Hence, indicator functions are binary functions that indicate if their argument is in the associated set or not. We use the notation $\mathbb{1}_{\mathcal{A}}$ to highlight the difference to the indicator function $\delta_{\mathcal{A}}$ previously defined in Equation 2.1. Using indicator functions of the form (2.4), simple functions are defined as follows:

Definition 2.1.30 (Simple Function) *Let $(\mathcal{S}, \mathfrak{A})$ be a measurable space. A map $f : \mathcal{S} \to \mathbb{R}$ is called a* simple function *if there exist $l \in \mathbb{N}$, mutually disjoint sets $\mathcal{A}_i \in \mathfrak{A}$, and real*

vectors $a_i \in \mathbb{R}^m$ for $i = 1, \dots, l$ such that

$$f(x) = \sum_{i=1}^{l} a_i \mathbb{1}_{\mathcal{A}_i}(x).$$

If $f, g : \mathcal{S} \to \mathbb{R}^m$ with $g_i(x) \leq f_i(x)$ for $i = 1, \dots, m$ and any $x \in \mathcal{S}$, then we write $g \leq f$. Likewise, we write the weaker condition $g \leq f$ a.e. if there exists a μ-null set N such that $g(x) \leq f(x)$ for any $x \in N^C$.

We denote by $\mathbb{S}$ the vector space of simple functions on $(\mathcal{S}, \mathfrak{A})$ and by $\mathbb{S}_+ := \{f \in \mathbb{S} : f \geq 0\}$ the space of nonnegative simple functions. Then, we define the mapping $I : \mathbb{S}_+ \to [0, \infty]^m$ by

$$I(f) = \sum_{i=1}^{l} a_i \mu(\mathcal{A}_i)$$

if the simple function f is represented by $f = \sum_{i=1}^{l} a_i \mathbb{1}_{\mathcal{A}_i}$. Compared to the simple functions, this mapping accounts for the measure μ of the associated set instead of the indicator function.

Next, we can define the Lebesgue integral for nonnegative functions based on simple functions and the previous mapping I.

Definition 2.1.31 (Lebesgue Integral) *If $f : \mathcal{S} \to [0, \infty]^m$ is measurable, we define the* Lebesgue integral *with respect to μ by*

$$\int_{\mathcal{S}} f \, \mathrm{d}\mu := \sup_{g \in \mathbb{S}_+ : g \leq f} I(g).$$

Figure 2.3 compares the classical definition of an integral due to Riemann with the Lebesgue integral. While the Riemann integral partitions the x-axis into equally spaced bins, the Lebesgue integral partitions the y-axis into equally spaced bins and approximates the functions by means of simple functions. Despite the different definition of the Riemann and Lebesgue integral, both integrals coincide in the limit for smooth functions integrated over a compact set [87, Section 4.3, Theorem 4.23]. In contrast, the non-smooth indicator function of the rational numbers $\mathbb{1}_{\mathbb{Q}}$ is Lebesgue integrable but not Riemann integrable [87].

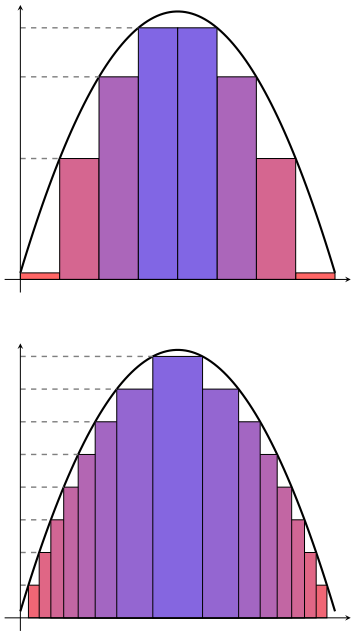


Figure 2.3: Visual comparison of the Riemann (top) and Lebesgue (bottom) integral.

Based on this definition of the Lebesgue integral for nonnegative functions, we introduce the property of μ-integrability and define the set of integrable functions.

Definition 2.1.32 (Integral of Measurable Functions) *Let $(\mathcal{S}, \mathfrak{A}, \mu)$ be a measure space and $(\mathcal{S}', \|\cdot\|)$ a Banach space for $\mathcal{S}' \subset \mathbb{R}^m$. A measurable function $f : \mathcal{S} \to \mathcal{S}'$ is called* μ-integrable *if*

$$\int_{\mathcal{S}} \|f\| \, \mathrm{d}\mu < \infty.$$

We define the set of these functions as

$$\mathrm{L}^1(\mathcal{S}, \mathfrak{A}, \mu) := \left\{ f : \mathcal{S} \to \mathcal{S}' : f \text{ is measurable and } \int_{\mathcal{S}} \|f\| \, \mathrm{d}\mu < \infty \right\}.$$

For any function $f \in \mathrm{L}^1(\mathcal{S}, \mathfrak{A}, \mu)$, we define the Lebesgue integral *of f w.r.t. μ by*

$$\int_{\mathcal{S}} f(x) \mu(\mathrm{d}x) := \int_{\mathcal{S}} f \, \mathrm{d}\mu := \begin{pmatrix} \int_{\mathcal{S}} \max(0, f_1) \, \mathrm{d}\mu \\ \vdots \\ \int_{\mathcal{S}} \max(0, f_m) \, \mathrm{d}\mu \end{pmatrix} - \begin{pmatrix} \int_{\mathcal{S}} -\max(0, -f_1) \, \mathrm{d}\mu \\ \vdots \\ \int_{\mathcal{S}} -\max(0, -f_m) \, \mathrm{d}\mu \end{pmatrix}.$$

Finally, we are able to define the Lebesgue space of integrable functions for a real number p.

Definition 2.1.33 (Lebesgue Space) *Let $(\mathcal{S}, \mathfrak{A}, \mu)$ be a measure space and let $(\mathcal{S}', \|\cdot\|)$ be a Banach space for $\mathcal{S}' \subset \mathbb{R}^m$. For a real number $1 \leq p \leq \infty$, the corresponding* Lebesgue space *is defined as*

$$\mathrm{L}^p(\mathcal{S}, \mathfrak{A}, \mu) \coloneqq \left\{ f : \mathcal{S} \to \mathcal{S}' : f \text{ measurable and } \left\|f\right\|_{\mathrm{L}^p(\mathcal{S},\mathfrak{A},\mu)} < \infty \right\},$$

with the norm defined for the case $1 \leq p < \infty$ as

$$\left\|f\right\|_{\mathrm{L}^p(\mathcal{S},\mathfrak{A},\mu)} \coloneqq \left(\int_{\mathcal{S}} \left\|f(x)\right\|^p \, \mathrm{d}\mu \right)^{\frac{1}{p}}$$

and for the case $p = \infty$ as

$$\left\|f\right\|_{\mathrm{L}^\infty(\mathcal{S},\mathfrak{A},\mu)} \coloneqq \inf_{N : N \subset \mathcal{S}, \mu(N) = 0} \left(\sup_{x \in \mathcal{S} \setminus N} \left\|f(x)\right\| \right).$$

We use the shorthand $\mathrm{L}^p(\mathcal{S}, \mathcal{S}') = \mathrm{L}^p(\mathcal{S}, \mathfrak{A}, \mathrm{L}^n)$ to denote measurable and Lebesgue-integrable functions $f : \mathcal{S} \to \mathcal{S}'$. If not specified otherwise, in this thesis we use the Lebesgue measure $\mu = \mathrm{L}^n$ on $\mathbb{R}^n$ and the shorthand $\mathrm{d}x = \mu(\mathrm{d}x) = \mathrm{L}^n(\mathrm{d}x)$.

The following theorem due to Fischer and Riesz shows that Lebesgue spaces are in fact Banach spaces.

Theorem 2.1.10 (Fischer–Riesz) *The Lebesgue space $\mathrm{L}^p(\mathcal{S}, \mathfrak{A}, \mu)$ for $1 \leq p \leq \infty$ is a Banach space.*

Proof. See [87, Section 7.3, Theorem 7.18]. □

The concept of duality can also be extended to Lebesgue spaces as the subsequent theorem shows.

Theorem 2.1.11 (Lebesgue Dual Space) *Let $p \in (1, \infty)$ and assume $\frac{1}{p} + \frac{1}{q} = 1$. Then, the dual space $(\mathrm{L}^p(\mathcal{S}, \mathfrak{A}, \mu))^*$ is given by $\mathrm{L}^q(\mathcal{S}, \mathfrak{A}, \mu)$.*

Proof. See [87, Section 7.6, Theorem 7.50]. □

Using the Lebesgue integral, we can relate measures, measurable functions and image measures in an integral equation.

Theorem 2.1.12 (Image Measure) *Let $(\mathcal{S}, \mathfrak{A})$ and $(\mathcal{S}', \mathfrak{A}')$ be measurable spaces, let μ be a measure on $(\mathcal{S}, \mathfrak{A})$ and let $X : \mathcal{S} \to \mathcal{S}'$ be measurable. Let $\mu' = \mu \circ X^{-1}$ be the image measure (or induced measure) of μ under the map X. Assume that $f : \mathcal{S}' \to \mathbb{R}$ is μ'-integrable, i.e. $f \in \mathrm{L}^1(\mathcal{S}', \mathfrak{A}', \mu')$. Then, $f \circ X \in \mathrm{L}^1(\mathcal{S}, \mathfrak{A}, \mu)$ and*

$$\int_{\mathcal{S}'} f \, \mathrm{d}\mu' = \int_{\mathcal{S}} (f \circ X) \, \mathrm{d}\mu = \int_{\mathcal{S}} f\left(X(x)\right) \mu(\mathrm{d}x).$$

Proof. See [87, Section 4.1, Theorem 4.10]. □

We will use this integral correspondence in the next section about probability theory.

To develop the Radon–Nikodym derivative, which we later require to link the cumulative distribution function with the density of a random variable, we first need to introduce the concept of absolute continuity to relate two measures.

Definition 2.1.34 (Absolute Continuity) *Let μ and ν be two measures on $(\mathcal{S}, \mathfrak{A})$. The measure ν is called* absolutely continuous *w.r.t. μ, denoted by $\nu \ll \mu$, if*

$$\mu(\mathcal{A}) = 0 \implies \nu(\mathcal{A}) = 0$$

for all $\mathcal{A} \in \mathfrak{A}$.

Finally, the Radon–Nikodym derivative follows from the next theorem.

Theorem 2.1.13 (Radon–Nikodym) *Let μ and ν be σ-finite measures on $(\mathcal{S}, \mathfrak{A})$ and let ν be absolutely continuous w.r.t. μ ($\nu \ll \mu$). Then, the function $f : \mathcal{S} \to \mathcal{S}$ is measurable in $\mathfrak{A}$ and finite a.e. and we have*

$$\nu(\mathcal{A}) = \int_{\mathcal{A}} f \, \mathrm{d}\mu = \int_{\mathcal{A}} \frac{\mathrm{d}\nu}{\mathrm{d}\mu} \, \mathrm{d}\mu$$

for any $\mathcal{A} \in \mathfrak{A}$. The function $\frac{\mathrm{d}\nu}{\mathrm{d}\mu}$ is called the Radon–Nikodym *derivative of ν w.r.t. μ and we write $f = \frac{\mathrm{d}\nu}{\mathrm{d}\mu}$.*

Proof. See [87, Section 7.4, Theorem 7.34]. □

As a result, the Radon–Nikodym derivative defines the density of the measure ν w.r.t. the measure μ.

2.2 Probability Theory

In this section, we recall basic results from probability theory collected from [85, 87].

The underlying idea of probability theory is to use *probability spaces* $(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ to model random experiments. A σ-algebra $\mathfrak{A} \subset 2^{\mathcal{S}}$ defines a class of sets whose elements are called *events* of $\mathcal{S}$. These events are observed by a *random variable* X which is a measurable mapping from $\mathcal{S}$ to a space of possible outcomes of a random experiment. The probabilities of the random outcomes are described by the distribution of the associated random variable, which is given by the image measure of $\mathbb{P}$ under X.

2.2.1 Probability Space

Let $\mathcal{S}$ be the space of elementary events, e.g. $\mathcal{S} \subset \mathbb{R}^n$, and let $\mathfrak{A} \subset 2^{\mathcal{S}}$ be a σ-algebra that defines the system of observable events $A \in \mathfrak{A}$. Then, this pair defines a measurable space as we have seen in Section 2.1.9. To account for the likelihood of events, we define the probability measure.

Definition 2.2.1 (Probability Measure) *Let $(\mathcal{S}, \mathfrak{A})$ be a measurable space and $\mu : \mathfrak{A} \to [0, \infty]$ a measure. Then, μ is a* probability measure *if $\mu(\mathcal{S}) = 1$.*

In the remainder, we denote probability measures by $\mathbb{P}$. Since probability measures are nonnegative and the probability of the entire set of events is $\mathbb{P}(\mathcal{S}) = 1$, we can deduce that $\mathbb{P} : \mathfrak{A} \to [0, 1]$.

A measurable space equipped with a probability measure defines a probability space, as stated in the next definition.

Definition 2.2.2 (Probability Space) *A triple* $(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ *is called a* probability space *if* $(\mathcal{S}, \mathfrak{A})$ *is a measurable space and* $\mathbb{P}$ *is a probability measure.*

In the following let $(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ be a complete probability space on $\mathcal{S} \subset \mathbb{R}^n$ and let $(\mathcal{S}', \mathfrak{A}')$ be a measurable space with $\mathcal{S}' \subset \mathbb{R}^m$.

2.2.2 Random Variables and their Characterization

Frequently, probabilistic events $A \in \mathfrak{A}$ on $\mathcal{S}$ cannot be observed directly. However, the effects of the underlying random events can be observed by means of measurable mappings from $\mathcal{S}$ to a space $\mathcal{S}'$. These random observations are measurable mappings and are called random variables.

Definition 2.2.3 (Random Variable) *Let* $X : \mathcal{S} \to \mathcal{S}'$ *be measurable. Then,* X *is called a* random variable *with values in* $(\mathcal{S}', \mathfrak{A}')$.

For an event $\mathcal{A}' \in \mathfrak{A}'$, we denote by $\{X \in \mathcal{A}'\} = \{s \in \mathcal{S} : X(s) \in \mathcal{A}'\} \coloneqq X^{-1}(\mathcal{A}')$ its pre-image of X and by $\mathbb{P}[X \in \mathcal{A}'] \coloneqq \mathbb{P}[X^{-1}(\mathcal{A}')]$ its image measure. Then, the probability that the random variable X takes on values in the event $\mathcal{A}'$ is given by

$$\mathbb{P}[X \in \mathcal{A}'] = \int_{X^{-1}(\mathcal{A}')} \mathrm{d}\mathbb{P}.$$

A random variable defines a distribution w.r.t. the probability measure $\mathbb{P}$ on its image space.

Definition 2.2.4 (Distribution) *Let* X *be a random variable. The probability measure* $\mathbb{P}_X \coloneqq \mathbb{P} \circ X^{-1}$ *is called the* distribution *of* X. *We write* $X \sim \mathcal{T}_{\mathcal{S}}$ *if* $\mathcal{T}_{\mathcal{S}} = \mathbb{P}_X$ *and say that the random variable is distributed by* $\mathcal{T}_{\mathcal{S}}$.

Thus, the distribution is the image measure of $\mathbb{P}$ under X. Next, we define the cumulative distribution function to characterize the distribution of a random variable.

Definition 2.2.5 (Cumulative Distribution Function) *Let* X *be a random variable. The map* $F_X : \mathcal{S} \to [0, 1], x \mapsto \mathbb{P}[X \leq x]$ *is called the* cumulative distribution function *of* X.

The cumulative distribution function is monotonically increasing and can be used to identify areas of high probability density since a high probability density implies a strong increase of the cumulative distribution function.

The pre-image of the cumulative distribution function $F_X : \mathcal{S} \to [0, 1]$ is used to define quantiles. Let $q \in (0, 1)$. The q^{th} *quantile* of the cumulative distribution function associated with the random variable X is defined as

$$F_X^{-1}(q) \coloneqq \min \left\{ x \in \mathcal{S} : F(x) \leq q \right\}.$$

So, it is the smallest value $x \in \mathcal{S}$ such that q percent of all values are below this value.

Definition 2.2.6 (Density Function) *If the cumulative distribution function $F_X : \mathbb{R}^n \to [0,1]$ is of the form*

$$F_X(x) = \int_{(-\infty,x]} p(t)\mu(\mathrm{d}t) = \int_{-\infty}^{x_1} \dots \int_{-\infty}^{x_n} p_X(x_1,\dots,x_n)\mu(\mathrm{d}x_1,\dots,\mathrm{d}x_n)$$

for all $x \in \mathbb{R}^n$ and some μ-integrable function $p_X : \mathbb{R}^n \to [0,\infty)$, then p_X is called the density *of the distribution of X w.r.t. the measure μ.*

In particular, let $X : \mathcal{S} \to \mathcal{S}'$ be a random variable with distribution $\mathbb{P}_X$. Let $\mathbb{P}_X$ be absolutely continuous w.r.t. the Lebesgue measure L^n, i.e. $\mathbb{P}_X \ll \mathrm{L}^n$. Then, the *density* of X w.r.t. the Lebesgue measure is given by the Radon–Nikodym derivative

$$p_X = \frac{\mathrm{d}\mathbb{P}_X}{\mathrm{dL}^n},$$

see Theorem 2.1.13. As a result, the probability that the random variable X takes in values in the event $\mathcal{A}'$ is given by

$$\mathbb{P}[X \in \mathcal{A}'] = \int_{X^{-1}(\mathcal{A}')} \mathrm{d}\mathbb{P} = \int_{\mathcal{A}'} p_X \,\mathrm{dL}^n.$$

As an example let us consider a random variable X that is defined over $\mathbb{R}^n$ and normally distributed, i.e. $X \sim \mathcal{N}(y,\Sigma)$ for a given mean $y \in \mathbb{R}^n$ and a covariance matrix $\Sigma \in \mathbb{R}^{n\times n}$. Then, its density w.r.t. L^n reads as

$$p_X(x) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\left(-\tfrac{1}{2}(x-y)^\top \Sigma^{-1}(x-y)\right)$$

and the cumulative distribution function is given by

$$\mathbb{P}[X \le x] = F_X(x) = \det(2\pi\Sigma)^{-1} \int_{(-\infty,x]} \exp\left(-\tfrac{1}{2}(s-y)^\top \Sigma^{-1}(s-y)\right) \mathrm{L}^n(\mathrm{d}s).$$

2.2.3 Independence of Random Variables

In many applications, we are interested in determining the probability of a family of random observations. This can be achieved by defining the dependence structure of the corresponding events and random variables. In this section, we discuss the concept of independence of events and random variables that allows for an efficient factorization of the joint probabilities.

To define the independence of a family of events, we have to ensure that for each subfamily of events the associated probability factorizes, which is summarized in the following definition.

Definition 2.2.7 (Independence of Events) *Let $\mathcal{I} \subset \mathbb{N}$ be an arbitrary index set and let $(\mathcal{A}_i)_{i\in\mathcal{I}}$ be an arbitrary family of events. The family $(\mathcal{A}_i)_{i\in\mathcal{I}}$ is called* independent *if for any finite subset $\mathcal{J} \subset \mathcal{I}$*

$$\mathbb{P}\left[\bigcap_{j\in\mathcal{J}} \mathcal{A}_j\right] = \prod_{j\in\mathcal{J}} \mathbb{P}[\mathcal{A}_j]$$

holds true.

To develop the independence of random variables, let us first define their joint distribution.

Definition 2.2.8 (Joint Distribution) *Let $\mathcal{I} \subset \mathbb{N}$ be an arbitrary index set and for all $i \in \mathcal{I}$, let X_i be a random variable. For any finite subset $\mathcal{J} \subset \mathcal{I}$, we define the* joint distribution function *of the family of random variables $(X_j)_{j\in\mathcal{J}}$ as*

$$F_{\mathcal{J}} \coloneqq F_{(X_j)_{j\in\mathcal{J}}} : S^{|\mathcal{J}|} \to [0,1],$$
$$x \mapsto \mathbb{P}\left[X_j \leq x_j \ \forall j \in \mathcal{J}\right] = \mathbb{P}\left[\bigcap_{j\in\mathcal{J}} X_j^{-1}\left((-\infty, x_j]\right)\right].$$

The probability measure $\mathbb{P}_{(X_j)_{j\in\mathcal{J}}}$ on $\mathbb{R}^{|\mathcal{J}|}$ is called the joint distribution *of $(X_j)_{j\in\mathcal{J}}$.*

From a computational point of view, evaluating the joint distribution function is challenging in high-dimensional spaces since the probability measure at the intersection of all pre-image sets of each random variable needs to be determined. However, for independent random variables, the cumulative distribution function factorizes into the product of the individual cumulative distribution functions as we see in the next theorem. This allows for a much more efficient evaluation of the joint distribution.

Theorem 2.2.1 (Independence of Random Variables) *A family of random variables $(X_i)_{i\in\mathcal{I}}$ for an arbitrary index set $\mathcal{I} \subset \mathbb{N}$ is* independent *if and only if for every finite $\mathcal{J} \subset \mathcal{I}$ and every $x = (x_j)_{j\in\mathcal{J}} \in \mathbb{R}^{|\mathcal{J}|}$*

$$F_{\mathcal{J}}(x) = \prod_{j\in\mathcal{J}} F_{\{j\}}(x_j).$$

holds true.

Proof. See [87, Section 2.2, Theorem 2.21]. □

If we further assume that F_J has a continuous density $p_J = p_{(X_j)_{j\in\mathcal{J}}}$, which is the *joint density* of the family of random variables, then also the joint density factorizes such that

$$p_{\mathcal{J}}(x) = \prod_{j\in\mathcal{J}} p_j(x_j)$$

holds for all $x = (x_j)_{j\in\mathcal{J}} \in \mathbb{R}^{|\mathcal{J}|}$.

To conclude this section, we define the notion of identically distributed random variables.

Definition 2.2.9 (Identically Distributed) *Let $\mathcal{I} \subset \mathbb{N}$ be an arbitrary index set. A family of random variables $(X_i)_{i\in\mathcal{I}}$ is called* identically distributed *if*

$$\mathbb{P}_{X_i} = \mathbb{P}_{X_j}$$

for all $i, j \in \mathcal{I}$.

As a shorthand we write that a family of random variables $(X_i)_{j\in\mathcal{I}}$ is independent and identically distributed (i.i.d.) if $(X_i)_{j\in\mathcal{I}}$ is independent and $\mathbb{P}_{X_i} = \mathbb{P}_{X_j}$ for all $i, j \in \mathcal{I}$.

2.2.4 Moments

The statistics of random variables are typically characterized by means of moments such as the expectation and variance.

Definition 2.2.10 (Expectation) *Let $X \in \mathrm{L}^1(\mathbb{P})$, we call*

$$\mathbb{E}[X] \coloneqq \int_{\mathcal{S}} X \, \mathrm{d}\mathbb{P}$$

the expectation *or* mean *of X. Moreover, for a measurable and integrable function $g : \mathcal{S}' \to \mathcal{S}''$ we define the expectation of g w.r.t. X as*

$$\mathbb{E}[g \circ X] = \int_{\mathcal{S}} g \circ X \, \mathrm{d}\mathbb{P} = \int_{\mathcal{S}'} g \, \mathrm{d}\mathbb{P}_X =: \mathbb{E}_{X \sim \mathcal{T}_{\mathcal{S}}}[g].$$

Thus, the expectation computes the mean of the entire space of elementary events w.r.t. the probability measure.

Next, we define the variance to characterize the deviation of a random variable from its expectation.

Definition 2.2.11 (Variance) *Let $X \in \mathrm{L}^2(\mathbb{P})$. Then, the* variance *of X is defined as*

$$\mathrm{Var}[X] \coloneqq \mathbb{E}[X^2] - \mathbb{E}[X]^2.$$

We denote by $\sigma \coloneqq \sqrt{\mathrm{Var}[X]}$ the standard deviation *of X.*

In the next theorem, we show important calculus rules regarding the expectation. In fact, we will see that the expectation is linear.

Theorem 2.2.2 (Calculus for Expectations) *Let $X, Y \in \mathrm{L}^1(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ be random variables on $(\mathcal{S}, \mathfrak{A}, \mathbb{P})$.*

- *If $\mathbb{P}_X = \mathbb{P}_Y$, then $\mathbb{E}[X] = \mathbb{E}[Y]$.*
- *Let $c \in \mathbb{R}$. Then $cX \in \mathrm{L}^1(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ and $X + Y \in \mathrm{L}^1(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ as well as*

$$\mathbb{E}[cX] = c\mathbb{E}[X] \quad \text{and} \quad \mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y].$$

- $|\mathbb{E}[X]| \leq \mathbb{E}[|X|]$.
- *If X, Y are independent, then $(XY) \in \mathrm{L}^1(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ and $\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y]$.*

Proof. See [87, Section 5.1 Theorem 5.3 and 5.4]. □

Let us conclude this section with some remarks on the computation of expectations for absolutely continuous and discrete probability measures.

First, we consider a probability measure $\mathbb{P}_X \ll \mathrm{L}^n$. Then, the expectation of a measurable and integrable function $g : \mathcal{S}' \to \mathcal{S}''$ can be computed by

$$E_{X \sim \mathcal{T}_{\mathcal{S}}}[g] = \int_{\mathcal{S}'} g \, \mathrm{d}\mathbb{P}_X = \int_{\mathcal{S}} g(x) p_X(x) \mathrm{L}^n(\mathrm{d}x),$$

where p_X is the density of $\mathbb{P}_X$ w.r.t. the Lebesgue measure L^n.

Second, we consider a discrete probability space. Let $\mathcal{S} = \{s_1, \ldots, s_n\}$ for $n \in \mathbb{N}$. Then, we define the discrete probability space $(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ by using a σ-algebra $\mathfrak{A} \subset 2^{\mathcal{S}}$ and the probability measure defined as

$$\mathbb{P} = \sum_{i=1}^{n} \alpha_i \mathbb{1}_{\{s_i\}}$$

for $\alpha_i \in \mathbb{R}_+$ and $\sum_{i=1}^n \alpha_i = 1$. Again we consider a random variable $X : \mathcal{S} \to \mathcal{S}'$ for some measurable space $(\mathcal{S}', \mathfrak{A}')$. Then, the expectation of a measurable function $g : \mathcal{S}' \to \mathcal{S}''$ is simply given by

$$\mathbb{E}_{x\sim\mathcal{T}_S}[g] = \sum_{s_i \in \mathcal{S}} g(s_i)\mathbb{P}(s_i) = \sum_{i=1}^{n} g(s_i)\alpha_i.$$

2.2.5 Conditional Probabilities

Conditional probabilities are a concept to compute the probabilities of events associated with a random experiment given some partial knowledge of the experiment's outcome. Throughout this section, we consider a probability space $(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ and a random variable $X \in \mathrm{L}^1(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ mapping to a measurable space $(\mathcal{S}', \mathfrak{A}')$. We denote the generated σ-algebra of X by $\sigma(X) = X^{-1}(\mathfrak{A}')$.

To define the conditional probability, we first need to introduce the conditional expectation.

Definition 2.2.12 (Conditional Expectation) *A random variable Y is called a* conditional expectation *of a random variable X given a σ-algebra $\mathfrak{F} \subset \mathfrak{A}$, denoted by $Y = \mathbb{E}[X|\mathfrak{F}]$, if*

- *Y is measurable w.r.t. $\mathfrak{F}$.*
- *For any $A \in \mathfrak{F}$, we have $\mathbb{E}[X\mathbb{1}_A] = \mathbb{E}[Y\mathbb{1}_A]$.*

Therefore, all equations involving conditional expectations are understood as equalities almost surely (a. s.), even if not explicitly stated. To interpret this rather abstract definition of the conditional expectation, we present the following corollary.

Corollary 2.2.3 (Conditional Expectation as Projection) *Let $\mathfrak{F} \subset \mathfrak{A}$ be a σ-algebra and let $X \in \mathrm{L}^2(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ with $\mathbb{E}[X^2] < \infty$. Then $\mathbb{E}[X|\mathfrak{F}]$ is the* orthogonal projection *of X onto $\mathrm{L}^2(\mathcal{S}, \mathfrak{F}, \mathbb{P})$. That is, for any measurable random variable $Y \in \mathrm{L}^2(\mathcal{S}, \mathfrak{F}, \mathbb{P})$ we have*

$$\mathbb{E}\left[(X - Y)^2\right] \geq \mathbb{E}\left[(X - \mathbb{E}[X|\mathfrak{F}])^2\right]$$

and equality holds if and only if $Y = \mathbb{E}[X|\mathfrak{F}]$.

Proof. See [87, Section 8.2, Corollary 8.17]. □

Thus, the conditional expectation is the orthogonal projection of a square integrable random variable onto the σ-algebra $\mathfrak{F}$.

Theorem 2.2.4 (Existence and Uniqueness of Conditional Expectation) *The conditional expectation $\mathbb{E}[X|\mathfrak{F}]$ exists and is unique up to equality a. s.*

Proof. See [87, Section 8.2, Theorem 8.12]. □

Based on this definition of the conditional expectation for events, we define the conditional expectation given a random variable Y as

$$\mathbb{E}[X|Y] := \mathbb{E}[X|\sigma(Y)].$$

In analogy to the calculus rules of the expectation, we present fundamental calculus rules for the conditional expectation in the following theorem.

Theorem 2.2.5 (Calculus for Conditional Expectations) *Let $\mathfrak{G} \subset \mathfrak{F} \subset \mathfrak{A}$ be σ-algebras and $Y \in \mathrm{L}^1(\mathcal{S}, \mathfrak{A}, \mathbb{P})$. Then:*

- $\mathbb{E}[\alpha X + Y|\mathfrak{F}] = \alpha\mathbb{E}[X|\mathfrak{F}] + \mathbb{E}[Y|\mathfrak{F}]$ *for any* $\alpha \in \mathbb{R}$.
- *If* $\mathbb{E}[|XY|] < \infty$ *and* Y *measurable w.r.t.* $\mathfrak{F}$*, then*

$$\mathbb{E}[XY|\mathfrak{F}] = Y\mathbb{E}[X|\mathfrak{F}] \quad \textit{and} \quad \mathbb{E}[Y|\mathfrak{F}] = \mathbb{E}[Y|Y] = Y.$$

- $\mathbb{E}[\mathbb{E}[X|\mathfrak{F}]|\mathfrak{G}] = \mathbb{E}[\mathbb{E}[X|\mathfrak{G}]|\mathfrak{F}] = \mathbb{E}[X|\mathfrak{G}]$
- $|\mathbb{E}[X|\mathfrak{F}]| \leq \mathbb{E}[|X|\,|\mathfrak{F}]$.
- *If* $\sigma(X)$ *and* $\mathfrak{F}$ *are independent, then* $\mathbb{E}[X|\mathfrak{F}] = \mathbb{E}[X]$.

Proof. See [87, Section 8.2 Theorem 8.14]. □

Finally, we are able to define the conditional probability:

Definition 2.2.13 (Conditional Probability) *Let $\mathcal{A} \in \mathfrak{A}$. The* conditional probability *of $\mathcal{A}$* given *the σ-algebra $\mathfrak{F}$ is defined as*

$$\mathbb{P}[\mathcal{A}|\mathfrak{F}] \coloneqq \mathbb{E}[\mathbb{1}_{\mathcal{A}}|\mathfrak{F}].$$

As a direct consequence of Theorem 2.2.5, we get that

$$\mathbb{P}[X|Y] = \mathbb{P}_X \quad \text{and} \quad \mathbb{P}[Y|X] = \mathbb{P}_Y$$

if and only if X, Y are independent random variables.

Next, we derive Bayes' theorem for discrete and continuous probability spaces. First, we consider the discrete set $\mathcal{S} = \mathcal{X} \times \mathcal{Y}$ with $\mathcal{X} = \{x_1, \dots, x_n\}$ and $\mathcal{Y} = \{y_1, \dots, y_m\}$ for $m, n \in \mathbb{N}$. We define the discrete probability space $(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ for a σ-algebra $\mathfrak{A} \subset 2^{\mathcal{S}}$ and the discrete probability measure

$$\mathbb{P} = \sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_{i,j}\mathbb{1}_{(x_j,y_i)}$$

for $\alpha_{i,j} \in \mathbb{R}_+$ such that $\sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_{i,j} = 1$. We consider two random variables $X, Y : \mathcal{S} \to \mathcal{S}'$ for some measurable space $(\mathcal{S}', \mathfrak{A}')$. The *marginal probability measures* of the two random variables are given by

$$\mathbb{P}_Y[y_i] = \sum_{j=1}^{n} \mathbb{P}[x_j, y_i] = \sum_{j=1}^{n} \alpha_{i,j} \quad \text{and} \quad \mathbb{P}_X[x_j] = \sum_{i=1}^{m} \mathbb{P}[x_i, y_j] = \sum_{i=1}^{m} \alpha_{i,j}$$

for $y_i \in \mathcal{Y}$ and $x_j \in \mathcal{X}$. Then, the *conditional probability measure* of the event $\{X = x\}$ given the event $\{Y = y\}$ for any $(x, y) \in \mathcal{S}$ is given by

$$\mathbb{P}[X = x|Y = y] \coloneqq \begin{cases} \frac{\mathbb{P}[x,y]}{\mathbb{P}_Y[y]} & \text{if } \mathbb{P}_Y[y] > 0, \\ 0 & \text{else} \end{cases}$$

and vice versa

$$\mathbb{P}[Y = y|X = x] \coloneqq \begin{cases} \frac{\mathbb{P}[x,y]}{\mathbb{P}_X[x]} & \text{if } \mathbb{P}_X[x] > 0, \\ 0 & \text{else.} \end{cases}$$

Thus, we have that the joint probability of X, Y factorizes such that

$$\mathbb{P}[x, y] = \mathbb{P}_{X|Y}[x, y]\mathbb{P}_Y[y] = \mathbb{P}_{Y|X}[y, x]\mathbb{P}_X[x]$$

for any $(x, y) \in \mathcal{S}$ with $\mathbb{P}_X[x], \mathbb{P}_Y[y] > 0$ using the shorthand notation $\mathbb{P}_{X|Y}[x, y] = \mathbb{P}[X = x|Y = y]$. Dividing both sides by $\mathbb{P}_Y[y]$ results in Bayes's theorem.

Theorem 2.2.6 (Bayes' Theorem — discrete) *Let $(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ be a discrete probability space and let the joint and marginal probability measures be defined as above. Then*

$$\mathbb{P}_{X|Y} = \frac{\mathbb{P}_{Y|X}\mathbb{P}_X}{\mathbb{P}_Y}.$$

Finally, we consider a continuous probability space over the set $\mathcal{S} \subset \mathbb{R}^l$. Let $\mathbb{P}$ be an absolutely continuous probability measure $\mathbb{P} \ll \mathrm{L}^l$ and $(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ the associated complete probability space. We again consider two random variables $X : \mathcal{S} \to \mathcal{X} \subset \mathbb{R}^n$ and $Y : \mathcal{S} \to \mathcal{Y} \subset \mathbb{R}^m$ for some measurable space $(\mathcal{X}, \sigma(X))$ and $(\mathcal{Y}, \sigma(Y))$ and denote the *joint density* of these random variables by $p_{X,Y} : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$. In analogy to the discrete case, the *marginal densities* are given by

$$p_X(x) = \int_{\mathcal{Y}} p_{X,Y}(x, y)\mathrm{L}^m(\mathrm{d}y) \quad \text{and} \quad p_Y(y) = \int_{\mathcal{X}} p_{X,Y}(x, y)\mathrm{L}^n(\mathrm{d}x).$$

Then, the *conditional densities* are given by

$$p_{X|Y}(x|y) \coloneqq \begin{cases} \frac{p(x,y)}{p_Y(y)} & \text{if } p_Y(y) > 0 \\ 0 & \text{else} \end{cases} \quad \text{and} \quad p_{Y|X}(y|x) \coloneqq \begin{cases} \frac{p(x,y)}{p_X(x)} & \text{if } p_X(x) > 0 \\ 0 & \text{else} \end{cases}$$

for any $(x, y) \in \mathcal{S}$. Consequently, the joint density factorizes into the conditional and marginal density, i.e.

$$p(x, y) = p_{X|Y}(x|y)p_Y(y) = p_{Y|X}(y|x)p_X(x)$$

for any $(x, y) \in \mathcal{S}$ with $p_X(x), p_Y(y) > 0$, which results in Bayes's theorem.

Theorem 2.2.7 (Bayes' Theorem — continuous) *Let $(\mathcal{S}, \mathfrak{A}, \mathbb{P})$ be a continuous probability space and let the joint and marginal probability densities be defined as above. Then*

$$p_{X|Y} = \frac{p_{Y|X}p_X}{p_Y}.$$

2.3 Ordinary Differential Equations

In this section, we define ordinary differential equations (ODEs) and recall basic results from ODE theory collected from [88].

Let $(\mathbb{V}, \|\cdot\|)$ be a Banach space, $\mathcal{S} \subset \mathbb{V}$ an open subset, and let $k \in \mathbb{N}$. We consider *ordinary differential equations* of the form

$$\frac{\mathrm{d}^k x}{\mathrm{d}t^k} = f\left(t, x, \frac{\mathrm{d}x}{\mathrm{d}t}, \ldots, \frac{\mathrm{d}^{k-1}x}{\mathrm{d}t^{k-1}}\right)$$

for the unknown function $x \in C^k(\mathcal{I}, \mathcal{S})$, where $\mathcal{I} \subset \mathbb{R}$ and

$$\frac{\mathrm{d}^j x}{\mathrm{d}t^j}$$

denotes its j^{th}-order temporal derivative. We use the shorthand $\dot{x} = \frac{\mathrm{d}x}{\mathrm{d}t}$ and $\ddot{x} = \frac{\mathrm{d}^2 x}{\mathrm{d}t^2}$ for the first and second temporal derivative. Here, $f \in C^0(\mathbb{R} \times \mathcal{S}^k, \mathbb{V})$ and the time t is frequently called the *independent* and x the *dependent* variable. The highest temporal derivative of x written on the left hand side defines the *order* of the differential equation. The ODE is called *autonomous* if the function f does not depend on the

time t. A solution of ODEs of this form is a function $\xi \in C^k(\mathcal{I}, \mathcal{S})$ for an interval $\mathcal{J} \subset \mathcal{I}$ characterized by

$$\frac{\mathrm{d}^k \xi}{\mathrm{d}t^k} = f\left(t, \xi, \frac{\mathrm{d}\xi}{\mathrm{d}t}, \ldots, \frac{\mathrm{d}^{k-1}\xi}{\mathrm{d}t^{k-1}}\right) \quad \forall t \in \mathcal{J}.$$

Definition 2.3.1 (First-order Initial Value Problem) *Let $f \in C^0(\mathcal{I} \times \mathcal{S}, \mathbb{V})$ for open sets $\mathcal{I} \subset \mathbb{R}$ and $\mathcal{S} \subset \mathbb{V}$. A first-order ODE with initial condition $x(t_0) = x_0 \in \mathcal{S}$ and $t_0 \in \mathcal{I}$ reads as*

$$\dot{x} = f(t, x), \quad x(t_0) = x_0 \tag{2.5}$$

and is called an initial value problem.

The integration w.r.t. t on both sides of Equation (2.5), yields the *integral representation* of the first-order initial value problem given by

$$x(t) = x_0 + \int_{t_0}^{t} f(s, x(s))\,\mathrm{d}s.$$

2.3.1 Existence of Solutions

In this section, we elaborate on the existence of solutions for the first-order initial value problem (2.5). First, we will show the local existence of solutions by the Picard–Lindelöf theorem and state requirements that ensure a global existence of solutions of the first-order initial value problem. Finally, we characterize the smoothness of the associated solutions.

To develop local solutions of the initial value problem (2.5), we need the concept of fixed points of a mapping.

Definition 2.3.2 (Fixed Point) *A* fixed point *of a mapping $K : \mathcal{S} \to \mathcal{S}$ is an element $x \in \mathcal{S} \subset \mathbb{V}$ such that $K(x) = x$.*

We use the notation $K^j(x)$ to indicate the recursive application of the mapping, i.e. $K^j(x) = K(K^{j-1}(x))$ with $K^0(x) = x$. Next, we define a local Lipschitz constant as a function of the time t, to estimate the maximal change of a solution due to the right-hand side f.

Definition 2.3.3 (Locally Lipschitz Continuous) *A function $f \in C(\mathcal{I} \times \mathcal{S}, \mathbb{V})$ is called* locally Lipschitz continuous *in the second argument around the initial condition x_0 if for every $t \in \mathcal{I}$ there exists a $\delta > 0$ such that the number*

$$L(t) = \sup_{x,y \in \overline{\mathcal{B}}(x_0,\delta),\ x \neq y} \frac{\left\|f(t,x) - f(t,y)\right\|}{\left\|x - y\right\|}$$

is finite.

Using the definition of the local Lipschitz constant, we can ensure the local existence of solutions of the initial value problem (2.5) by the following theorem.

Theorem 2.3.1 (Picard–Lindelöf) *Suppose $f \in C^0(\mathcal{I} \times \mathcal{S}, \mathbb{R}^n)$, where $\mathcal{I} \subset \mathbb{R}$ and $\mathcal{S} \subset \mathbb{R}^n$ are open subsets. In addition, let f be locally Lipschitz continuous in the second argument. For initial values $t_0 \in \mathcal{I}$ and $x_0 \in \mathcal{S}$ choose $\delta, T > 0$ such that $[t_0, t_0 + T] \subset \mathcal{I}$*

and $\overline{\mathcal{B}}(x_0, \delta) \subset \mathcal{S}$. *Set*

$$M(t) = \int_{t_0}^{t} \sup_{x \in \overline{\mathcal{B}}(x_0, \delta)} \left\| f(s, x) \right\| \, \mathrm{d}s,$$

$$L(t) = \sup_{x, y \in \overline{\mathcal{B}}(x_0, \delta),\ x \neq y} \frac{\left\| f(t, x) - f(t, y) \right\|}{\left\| x - y \right\|}.$$

Thus, $M(T)$ *is non-decreasing and we define* T_0 *via*

$$T_0 = \sup \left\{ 0 < t \leq T : M(t_0 + t) \leq \delta \right\}.$$

Suppose

$$L_1(T_0) \coloneqq \int_{t_0}^{t_0 + T_0} L(t) \, \mathrm{d}t < \infty.$$

Then, the unique local solution $\widehat{x}(t)$ *of the initial value problem* (2.5) *is given by*

$$\widehat{x} = \lim_{i \to \infty} K^i(x_0) \in C^1([t_0, t_0 + T_0], \overline{\mathcal{B}}(x_0, \delta))$$

for the operator $K(x)(t) = x_0 + \int_{t_0}^{t} f(s, x(s)) \, \mathrm{d}s$, *and the estimate*

$$\sup_{t_0 \leq t \leq T_0} \left\| \widehat{x}(t) - K^i(x_0)(t) \right\| \leq \frac{L_1(T_0)^i}{i!} \exp(L_1(T_0)) \int_{t_0}^{t_0 + T_0} \left\| f(s, x_0) \right\| \, \mathrm{d}s$$

holds true. The same result holds true for $t < t_0$.

Proof. See [88, Section 2.3, Theorem 2.5]. □

In addition to the local existence of solutions, the Picard–Lindelöf theorem states that the local solution is unique and determined by a fixed point of the mapping K. Computing the solution by iteratively applying K is known as *Picard iteration*. However, this scheme is frequently not feasible since the integral in the definition of K cannot be solved in general.

To extend the existence of local solutions to all time values $t \in \mathbb{R}$, one needs to show that the growth defined by the right-hand side of the ODE of the form (2.5) is at most affine, as stated in the following theorem.

Theorem 2.3.2 (Extension of Local Solutions) *Let* $f \in C^0(\mathbb{R} \times \mathbb{V}, \mathbb{V})$ *and for every* $T > 0$ *there exist constants* $M(T)$ *and* $L(T)$ *such that*

$$\left\| f(t, x) \right\| \leq M(T) + L(T) \left\| x \right\|$$

holds for all $t \in [-T, T]$ *and* $x \in \mathbb{V}$. *Then, all solutions of the initial value problem* (2.5) *are defined for all* $t \in \mathbb{R}$.

Proof. See [88, Section 2.6, Theorem 2.17]. □

The Picard–Lindelöf theorem ensures that for $f \in C^0(\mathcal{I} \times \mathcal{S}, \mathbb{V})$ the solutions of the initial value problem (2.5) have an additional degree of smoothness and are in $C^1([t_0, t_0 + T_0, \overline{\mathcal{B}}(x_0, \delta)])$ locally. In fact, frequently the right-hand side f is differentiable, i.e. $f \in C^1(\mathcal{I} \times \mathcal{S}, \mathbb{V})$. Then, the following lemma states that the local solution of the initial value problem (2.5) has an additional degree of smoothness.

Lemma 2.3.3 (Smoothness of Solutions) *Let $f \in C^k(\mathcal{I} \times \mathcal{S}, \mathbb{V})$ for $k \geq 1$ and both $\mathcal{I} \subset \mathbb{R}$ and $\mathcal{S} \subset \mathbb{V}$ are open sets. Suppose $t_0 \in \mathcal{I}$ and $x_0 \in \mathcal{S}$. Then, the local solution of the initial value problem* (2.5) *is in $C^{k+1}(\mathcal{I}, \mathcal{S})$.*

Proof. See [88, Section 2.2 Lemma 2.3]. □

2.3.2 Dependence on the Initial Condition

In many applications, the initial value $x_0 \in \mathbb{V}$ is degraded due to measurement uncertainties. Thus, it is very important to analyze the effect of perturbations of the initial value on the solution of an ODE of the form (2.5). Gronwall's theorem provides the necessary tools.

To develop Gronwall's theorem, we need to define a local Lipschitz constant of f that holds globally at any time t.

Definition 2.3.4 *Let $f \in C^0(\mathcal{I} \times \mathbb{V}, \mathbb{R}^n)$. f is called* locally Lipschitz continuous in the second argument, uniformly w.r.t. the first, *if for every compact set $\mathcal{J} \subset \mathcal{I}$ and $\mathcal{U} \subset \mathbb{V}$*

$$L = \sup_{t \in \mathcal{J},\, x,y \in \mathcal{U},\, x \neq y} \frac{\left\| f(t,x) - f(t,y) \right\|}{\left\| x - y \right\|}$$

is finite.

Using this definition, we can state Gronwall's theorem, also known as Gronwall's inequality.

Theorem 2.3.4 (Gronwall) *Let $f, g \in C^0(\mathcal{I} \times \mathbb{V}, \mathbb{R}^n)$ and let f be locally Lipschitz continuous in the second argument, uniformly w.r.t. the first. If $x(t)$ and $y(t)$ are the respective solutions of the initial value problems*

$$\dot{x} = f(t,x), \quad x(t_0) = x_0 \qquad \text{and} \qquad \dot{y} = g(t,y), \quad y(t_0) = y_0,$$

then

$$\left\| x(t) - y(t) \right\| \leq \left\| x_0 - y_0 \right\| \exp\left(L|t - t_0|\right) + \frac{M}{L} \left(\exp\left(L|t - t_0|\right) - 1 \right)$$

holds true, where

$$L = \sup_{t \in \mathcal{J},\, x,y \in \mathcal{S},\, x \neq y} \frac{\left\| f(t,x) - f(t,y) \right\|}{\left\| x - y \right\|}, \quad M = \sup_{t \in \mathcal{J},\, x \in \mathcal{S}} \left\| f(t,x) - g(t,x) \right\|$$

with $\mathcal{J} \subset \mathcal{I}$ and $\mathcal{S} \subset \mathbb{V}$ being sets containing the graphs of $x(t)$ and $y(t)$.

Proof. See [88, Section 2.4, Theorem 2.8]. □

As a result, Gronwall's theorem states that the deviation of two solutions of first-order initial value problems is due to different initial conditions (first term) and due to variations of the right-hand side (second term). Note the exponential influence of the Lipschitz constant L in both terms. Thus, the deviation of two solutions of the initial value problem (2.5) depends to a large extent on the Lipschitz constant.

2.4 Optimization

Optimization is an important mathematical discipline that is applied in numerous fields ranging from physics, engineering, machine learning, computer vision, and medicine to finance [84, 89, 90]. In addition, many natural systems seem to obey the laws of optimization. For example, a rolling ball on a hill will always end up at the bottom of the hill, which is the position with *minimal* potential energy. Due to the ubiquitous applicability of optimization methods, this field has been studied well. In this section, we review important concepts and results from optimization taken from [84, 89–92].

From a mathematical point of view an optimization problem is formally defined as a minimization problem of the form

$$\inf_{x \in \mathcal{S}} f(x), \tag{2.6}$$

where $\mathcal{S} \subset \mathbb{R}^n$ is called the *constraint set* defining all feasible solutions and $f : \mathcal{S} \to \overline{\mathbb{R}}$ is the *objective function* which assigns an energy value to each feasible solution $x \in \mathcal{S}$. Since we aim to minimize the objective function, we are searching for a solution with the smallest energy within the constraint set. Thus, an *optimal solution* $x^* \in \mathcal{S}$ of this problem fulfills

$$f(x^*) \leq f(x)$$

for all $x \in \mathcal{S}$.

The minimization problem (2.6) can be classified into different categories based on the constraint set and the smoothness of the objective function. If the constraint set is equal to the entire underlying space, i.e. $\mathcal{S} = \mathbb{V}$, (2.6) is called an *unconstrained optimization problem*, while a *constrained optimization problem* implies that $\mathcal{S} \subsetneq \mathbb{V}$. The optimization problem (2.6) is called *smooth* if the objective function f is continuously differentiable. If the objective function f is non-differentiable, problem (2.6) is called a *non-smooth optimization problem*.

2.4.1 Optimality Conditions

In this section, we state optimality conditions that characterize minimal and maximal points of a function. Throughout this chapter let $(\mathbb{V}, \|\cdot\|)$ be a Banach space over the field $\mathbb{R}$.

Definition 2.4.1 (Global Minimum and Maximum) *Let $f : \mathcal{S} \to \overline{\mathbb{R}}$ be defined over a set $\mathcal{S} \subset \mathbb{V}$. Then*

- *$x^* \in \mathcal{S}$ is called a* global minimum (maximum) *of f over $\mathcal{S}$ if $f(x^*) \leq f(x)$ ($f(x^*) \geq f(x)$) for all $x \in \mathcal{S}$.*
- *$x^* \in \mathcal{S}$ is called a* strict global minimum (maximum) *of f over $\mathcal{S}$ if $f(x^*) < f(x)$ ($f(x^*) > f(x)$) for all $x \in \mathcal{S}$, $x \neq x^*$.*

In contrast to a global minimum/maximum the optimality for local extremal points only holds in a local neighborhood, as we will see in the next definition.

Definition 2.4.2 (Local Minimum and Maximum) *Let $f : \mathcal{S} \to \overline{\mathbb{R}}$ be defined over a set $\mathcal{S} \in \mathbb{V}$. Then*

- *$x^* \in \mathcal{S}$ is called a* local minimum (maximum) *of f over $\mathcal{S}$ if there exists an $\epsilon > 0$ such that $f(x^*) \leq f(x)$ ($f(x^*) \geq f(x)$) for all $x \in \mathcal{S} \cap \mathcal{B}_{\|\cdot\|}(x^*, \epsilon)$.*
- *$x^* \in \mathcal{S}$ is called a* strict local minimum (maximum) *of f over $\mathcal{S}$ if there exists an*

$\epsilon > 0$ *such that* $f(x^*) < f(x)$ $(f(x^*) > f(x))$ *for all* $x \in \mathcal{S} \cap \mathcal{B}_{\|\cdot\|}(x^*, \epsilon)$, $x \neq x^*$.

To illustrate the difference between global and local minima of a function, we show a plot of an example function in Figure 2.4. We analyze this one-dimensional function in the interval $\mathcal{S} = [-1, 8] \subset \mathbb{R}$. Within this interval, the plotted function has two global minima ($x = 3$ and $x = 6.5$) and infinitely many local minima in the interval $[1, 2]$. Note that the above definitions imply that every global minimum is also a local minimum. However, the converse is not necessarily true.

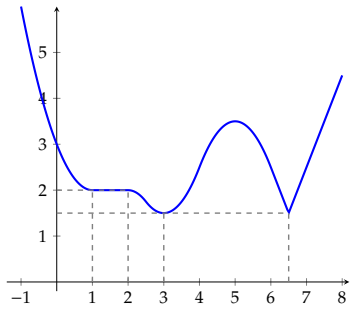


Figure 2.4: Visualization of different minima types.

To develop the first- and second-order optimality conditions, we first need to define the gradient and the Hessian of a function. Let $f \in C^k(\mathcal{S}, \overline{\mathbb{R}})$ for $k \geq 2$ and $\mathcal{S} \subset \mathbb{V}$ a subset of a finite-dimensional vector space. We define the gradient of f as

$$\nabla f(x) = \begin{pmatrix} \frac{\partial f}{\partial x_1}(x) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x) \end{pmatrix}$$

for $x = (x_1\ x_2\ \cdots\ x_n)^\top \in \mathbb{V}$. Then, we have that $\nabla f \in C^{k-1}(\mathcal{S}, \overline{\mathbb{R}}^n)$. Intuitively, the gradient $\nabla f(x) \in \overline{\mathbb{R}}^n$ at $x \in \mathcal{S}$ defines the direction which locally leads to the largest increase of f. In analogy to the gradient, we define the Hessian of a function as

$$\nabla^2 f(x) = \begin{pmatrix} \frac{\partial^2 f}{\partial^2 x_1}(x) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(x) & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(x) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(x) & \frac{\partial^2 f}{\partial^2 x_2}(x) & \vdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1}(x) & \frac{\partial^2 f}{\partial x_n \partial x_2}(x) & \cdots & \frac{\partial^2 f}{\partial^2 x_n}(x) \end{pmatrix}$$

and obtain that $\nabla^2 f \in C^{k-2}(\mathcal{S}, \mathbb{R}^{n \times n})$. While the gradient of a function can be used to identify directions of maximal increase, the Hessian $\nabla^2 f(x)$ describes the local curvature of f at x. Note that the Hessian $\nabla^2 f(x) \in \mathbb{R}^{n \times n}$ is a symmetric matrix at every point x since the order of differentiation does not matter [93, Satz von Schwarz].

The next theorem due to Fermat defines a necessary local optimality condition based on a function's gradient.

Theorem 2.4.1 (Fermat's First-order Optimality Condition) *Let* $f \in C^1(\mathcal{S}, \overline{\mathbb{R}})$ *for* $\mathcal{S} \subset \mathbb{V}$. *Suppose that* $x^* \in \operatorname{int}(\mathcal{S})$ *is a local optimum point. Then* $\nabla f(x^*) = 0$.

Proof. See [90, Section 2.1, Theorem 2.6]. □

Unfortunately, this theorem only provides a necessary condition for local optimum points and does not distinguish between minima, maxima, or saddle points (see Definition 2.4.5). Therefore, we define points of vanishing gradient as stationary points in the following theorem.

Definition 2.4.3 (Stationary Points) *Let* $f \in C^1(\mathcal{S}, \overline{\mathbb{R}})$ *for* $\mathcal{S} \subset \mathbb{V}$. *Let* $x^* \in \operatorname{int}(\mathcal{S})$. *Then* x^* *is called a* stationary point *of* f *if* $\nabla f(x^*) = 0$.

To check whether a stationary point is a minimum or maximum we need to evaluate the local curvature encoded in the Hessian at the stationary point. Before we can state the corresponding second-order optimality condition, we need to introduce a classification of symmetric matrices based on their definiteness.

Definition 2.4.4 (Definiteness of Matrices) *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix. A matrix A is called*

- positive semidefinite *if $x^\top Ax \geq 0$ for all $x \in \mathbb{R}^n \setminus \{0\}$, denoted by $A \succcurlyeq 0$.*
- negative semidefinite *if $x^\top Ax \leq 0$ for all $x \in \mathbb{R}^n \setminus \{0\}$, denoted by $A \preccurlyeq 0$.*
- indefinite *if there exist $x, y \in \overline{\mathbb{R}}^n$ such that $x^\top Ax > 0$ and $y^\top Ay < 0$.*

We omit the prefix 'semi' if the inequalities are strict and remove the lower equality line in the corresponding symbols, i.e $\succ$ and $\prec$.

A practically more useful characterization of the definiteness of matrices is based on their corresponding eigenvalues.

Theorem 2.4.2 (Eigenvalue Characterization of Matrices) *Let $A \in \mathbb{R}^{n\times n}$ be a matrix. We denote by $\lambda(A)_i$ for $i = 1, \ldots, n$ its eigenvalues. Then A is*

- positive semidefinite *if and only if all its eigenvalues $\lambda(A)_i \geq 0$ for $i = 1, \ldots, n$.*
- negative semidefinite *if and only if all its eigenvalues $\lambda(A)_i \leq 0$ for $i = 1, \ldots, n$.*
- indefinite *if and only if it has at least one positive eigenvalue and one negative eigenvalue.*

As before we omit the prefix 'semi' if the inequalities are strict.

Proof. See [90, Section 2.2, Theorem 2.17]. □

Using this characterization of matrices, we state the second-order necessary condition for local minima and maxima in the subsequent theorem.

Theorem 2.4.3 (Necessary Second-order Optimality Condition) *Let $f \in C^2(\mathcal{S}, \overline{\mathbb{R}})$ for an open set $\mathcal{S} \subset \mathbb{V}$. Suppose that $x^* \in \mathcal{S}$ is a stationary point. Then the following holds:*

- *If x^* is a local minimum of f over $\mathcal{S}$, then $\nabla^2 f(x^*) \succcurlyeq 0$.*
- *If x^* is a local maximum of f over $\mathcal{S}$, then $\nabla^2 f(x^*) \preccurlyeq 0$.*

Proof. See [90, Section 2.3, Theorem 2.26]. □

Like Fermat's first-order optimality condition, the latter theorem only provides a necessary condition. However, we can state a second-order sufficient optimality condition for *strict* local extremal points.

Theorem 2.4.4 (Sufficient Second-order Optimality Condition) *Let $f \in C^2(\mathcal{S}, \overline{\mathbb{R}})$ for an open set $\mathcal{S} \subset \mathbb{V}$. Suppose that $x^* \in \mathcal{S}$ is a stationary point. Then the following holds:*

- *If $\nabla^2 f(x^*) \succ 0$, then x^* is a strict local minimum of f over $\mathcal{S}$.*
- *If $\nabla^2 f(x^*) \prec 0$, then x^* is a strict local maximum of f over $\mathcal{S}$.*

Proof. See [90, Section 2.3, Theorem 2.27]. □

Note that this condition is only sufficient to characterize strict local extremal points but not necessary. For example, the function $f(x) = x^4$ has a strict global minimum at $x = 0$ but $f''(x) = 12x^2$ vanishes at $x = 0$.

The following definition introduces saddle points to characterize all stationary points that cannot be described by local minima or local maxima.

Definition 2.4.5 (Saddle Point) *Let $f \in C^1(\mathcal{S}, \overline{\mathbb{R}})$ for an open set $\mathcal{S} \subset \mathbb{V}$. A stationary point $x^* \in \mathcal{S}$ is called a* saddle point *of f over $\mathcal{S}$ if it is neither a local minimum nor a local maximum of f over $\mathcal{S}$.*

We can use the curvature encoded in the local Hessian to state a sufficient condition for a saddle point.

Theorem 2.4.5 (Sufficient Condition for a Saddle Point) *Let $f \in C^2(\mathcal{S}, \overline{\mathbb{R}})$ for an open set $\mathcal{S} \subset \mathbb{V}$. Suppose that $x^* \in \mathcal{S}$ is a stationary point. If $\nabla^2 f(x^*)$ is an indefinite matrix, then x^* is a saddle point of f over $\mathcal{S}$.*

Proof. See [90, Section 2.3, Theorem 2.29]. □

So far the first- and second-order optimality conditions only used local information extracted from the gradient and Hessian at a single point and thus lead only to local optimality conditions. As a consequence, optimality conditions that ensure global optimality of points must be based on global information. For example, if the Hessian of a function is positive semidefinite at all points, the curvature does not change its sign, which implies that every stationary point is a global minimum. In the next section, we call this property of a function *convexity*.

Theorem 2.4.6 (Global Optimality Condition) *Let $f \in C^2(\mathbb{V}, \mathbb{R})$. Suppose that $\nabla^2 f(x) \succ 0$ for all $x \in \mathbb{V}$. Let $x^* \in \mathbb{V}$ be a stationary point of f. Then x^* is a global minimum point of f.*

Proof. See [90, Section 2.3, Theorem 2.38]. □

Later, we will see that this optimality condition originates from the second-order definition of convex functions.

2.4.2 Convex Optimization

As the last optimality condition in the previous section suggests, the convexity of a function is an important property since every local minimum is a global minimum. Therefore, we discuss the area of *convex* optimization in this section. We first define convex sets and functions, then we introduce the concept of subgradients to deal with non-differentiable functions, and finally we introduce the convex conjugate and the proximal mapping.

2.4.2.1 Convex Sets

A fundamental building block of convex optimization are convex sets. They are formally defined as:

Definition 2.4.6 (Convex Set) *A set $\mathcal{S} \subset \mathbb{V}$ is called* convex *if*

$$\alpha x + (1 - \alpha) y \in \mathcal{S}$$

holds for any $x, y \in \mathcal{S}$ and $\alpha \in [0, 1]$.

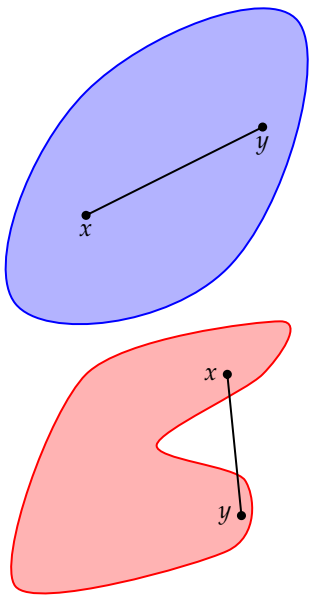


Figure 2.5: Visualization of a convex set (top) and a non-convex set (bottom).

This definition basically states that a convex set contains all line segments between its elements. Figure 2.5 illustrates a convex and a non-convex set. For the non-convex set, the black line depicts a counter-example, which does not fulfill the definition of a convex set.

In the subsequent theorem we present basic convex sets:

Theorem 2.4.7 (Basic Convex Sets) *Let $a \in \mathbb{V}$ and $b \in \mathbb{R}$. The following sets are convex:*

- *a* hyperplane $\mathcal{S} = \{x \in \mathbb{V} : \langle a, x \rangle = b\}$,
- *a* half-space $\mathcal{S} = \{x \in \mathbb{V} : \langle a, x \rangle \leq b\}$,
- *the* open ball $\mathcal{B}_{\|\cdot\|}(a, b)$ *and the* closed ball $\overline{\mathcal{B}}_{\|\cdot\|}(a, b)$ *for* $b > 0$.

Proof. See [90, Section 6.3, Lemma 6.3, Theorem 6.4]. □

To effectively work with convex sets, it is essential to understand which operations are convexity-preserving.

Theorem 2.4.8 (Convexity-preserving Operations) *Let $\mathcal{S}_i \subset \mathbb{V}$ be convex sets for any $i \in \mathcal{I}$, where $\mathcal{I} \subset \mathbb{N}$ is an index set.*

- *The* intersection *of convex sets $\bigcap_{i \in \mathcal{I}} \mathcal{S}_i$ is convex.*
- *Let $\alpha_i \in \mathbb{R}$ for $i \in \mathcal{I}$. Then the* weighted sum *of convex sets*

$$\sum_{i \in \mathcal{I}} \alpha_i \mathcal{S}_i := \left\{ \sum_{i \in \mathcal{I}} \alpha_i x_i : x_i \in \mathcal{S}_i \text{ for } i \in \mathcal{I} \right\}$$

  is convex.
- *Let $I = \{1, \ldots, m\}$. Then the* Cartesian product *of convex sets*

$$\mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \times \mathcal{S}_m := \{(x_1, x_2, \ldots, x_m) : x_i \in \mathcal{S}_i \text{ for } i \in \mathcal{I}\}$$

  is convex.

Let $\mathrm{A} : \mathbb{R}^n \to \mathbb{R}^m$ be a linear transform defined as $\mathrm{A}(x) = Ax + b$ for a matrix $A \in \mathbb{R}^{m \times n}$ and a vector $b \in \mathbb{R}^m$.

- *Let $\mathcal{S} \subset \mathbb{R}^n$ be a convex set. The* image *of a linear transform $\mathrm{A}(\mathcal{S}) := \{y \in \mathbb{R}^m : y = \mathrm{A}(x) \text{ for } x \in \mathcal{S}\}$ is convex.*
- *Let $\mathcal{S} \subset \mathbb{R}^m$ be a convex set. The* pre-image *of a linear transform $\mathrm{A}^{-1}(\mathcal{S}) := \{x \in \mathbb{R}^n : \mathrm{A}(x) \in \mathcal{S}\}$ is convex.*

Proof. See [92, Section 2.2.3, Theorem 2.2.8]. □

2.4.2.2 Convex Functions

Another building block of convex optimization are convex functions defined on top of convex sets.

Definition 2.4.7 (Convex Function) *A function $f : \mathcal{S} \to \overline{\mathbb{R}}$ defined over a convex set $\mathcal{S} \subset \mathbb{V}$ is called* convex *if*

$$f(\alpha x + (1 - \alpha) y) \leq \alpha f(x) + (1 - \alpha) f(y)$$

for all $x, y \in \mathcal{S}$ and $\alpha \in [0, 1]$.

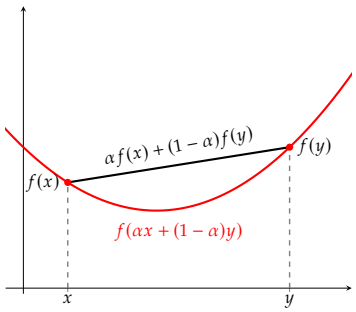


Figure 2.6: Visualization of the secant equation of a convex function.

Intuitively speaking, on any interval, a convex function is always below the secant spanned by the function values at the boundary of the interval, as illustrated in Figure 2.6.

This interpretation of convex functions motivates the following theorem that relates convex sets and convex functions by means of the epigraph.

Theorem 2.4.9 (Convexity of Epigraph) *A function $f : \mathcal{S} \to \overline{\mathbb{R}}$ for $\mathcal{S} \subset \mathbb{V}$ is convex if and only if its epigraph* $\operatorname{epi}(f) = \{(x, y) \in \mathbb{V} \times \mathbb{R} : f(x) \leq y\}$ *is convex.*

Proof. See [92, Section 3.1.1, Theorem 3.1.2]. □

An important class of functions in optimization are norms since they induce a distance metric on a space. In the next proposition, we show that every norm is in fact convex.

Proposition 2.4.10 (Norms are Convex) *Let $f(x) = \|x\|$ be any norm on $\mathbb{V}$. Then f is convex.*

Proof. For any $x, y \in \mathbb{V}$ and $\alpha \in [0, 1]$ we have:

$$\begin{aligned} f(\alpha x + (1-\alpha)y) &= \left\|\alpha x + (1-\alpha)y\right\| \\ &\leq \|\alpha x\| + \left\|(1-\alpha)y\right\| = \alpha \|x\| + (1-\alpha)\left\|y\right\| \\ &= \alpha f(x) + (1-\alpha) f(y). \end{aligned}$$

□

Let $m \in \mathbb{N}$. We call the weighted sum of points $x_1, \ldots, x_m \in \mathcal{S}$

$$x = \sum_{i=1}^{m} \alpha_i x_i$$

a *convex combination* if $\alpha \in \Delta_m$, where $\Delta_m \subset \mathbb{R}^m$ is the unit simplex defined as

$$\Delta_m := \left\{ \alpha \in \mathbb{R}^m : \sum_{i=1}^{m} \alpha_i = 1, \alpha_i \geq 0 \text{ for } i = 1, \ldots, m \right\}.$$

Then, the definition of convex functions states that the function value of a convex combination of two ($m = 2$) points is below the convex combination of the corresponding function values. Interestingly, this property can be extended to any finite number of points leading to Jensen's inequality.

Theorem 2.4.11 (Jensen's Inequality) *Let $f : \mathcal{S} \to \overline{\mathbb{R}}$ be a convex function over a convex set $\mathcal{S}$ and $m \in \mathbb{N}$. Then for any $x_1, \ldots, x_m \in \mathcal{S}$ and $\alpha \in \Delta_m$, the following inequality holds:*

$$f\left(\sum_{i=1}^{m} \alpha_i x_i\right) \leq \sum_{i=1}^{m} \alpha_i f(x_i).$$

Proof. See [90, Section 7.1, Theorem 7.5]. □

Next, we state the first-order condition of convexity for continuously differentiable functions.

Theorem 2.4.12 (First-order Condition of Convexity) *Let $f \in C^1(\mathcal{S}, \overline{\mathbb{R}})$ over a convex set $\mathcal{S} \subset \mathbb{V}$. Then f is convex if and only if*

$$f(y) \geq f(x) + \left\langle \nabla f(x), y - x \right\rangle \tag{2.7}$$

for all $x, y \in \mathcal{S}$.

Proof. See [90, Section 7.2, Theorem 7.6]. □

This condition has the neat geometric interpretation that the function value of a convex function always lies above any tangent, as depicted in Figure 2.7. Moreover, a continuously differentiable convex function f can be globally bounded from below by its first-order Taylor approximation.

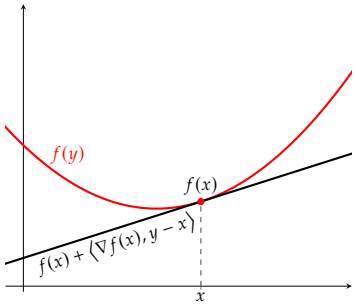


Figure 2.7: Visualization of the first-order condition of convexity.

If a function is twice continuously differentiable, the second-order condition of convexity, defined in the following theorem, can be used to check whether it is convex.

Theorem 2.4.13 (Second-order Condition of Convexity) *Let $f \in C^2(\mathcal{S}, \overline{\mathbb{R}})$ over an open convex set $\mathcal{S} \subset \mathbb{V}$. Then, f is convex if and only if*

$$\nabla^2 f(x) \succcurlyeq 0$$

for all $x \in \mathcal{S}$.

Proof. See [90, Section 7.3, Theorem 7.12]. □

Thus, a function $f \in C^2(\mathbb{V}, \overline{\mathbb{R}})$ is convex if its Hessian is positive semidefinite at every point in $\mathbb{V}$.

We conclude this section on convex functions by showing that the positively weighted sum of convex functions as well as a linear transformation of the argument of a convex function are convexity-preserving.

Theorem 2.4.14 (Positively Weighted Sum) *Let $f_1, \ldots, f_m : \mathcal{S} \to \overline{\mathbb{R}}$ be convex functions defined over a convex set $\mathcal{S}$ and $\alpha \in \mathbb{R}_+^m$. Then, the function*

$$f(x) = \sum_{i=1}^{m} \alpha_i f_i(x)$$

is convex.

Proof. See [90, Section 7.4, Theorem 7.16]. □

Theorem 2.4.15 (Linear Transform) *Let $f : \mathcal{S} \to \mathbb{R}$ be a convex function defined over a convex set $\mathcal{S} \subset \mathbb{R}^m$. Let $\mathrm{A} : \mathbb{R}^n \to \mathbb{R}^m$ be a linear transform. Then the function $g : \mathbb{R}^n \to \mathbb{R}$ defined as*

$$g(x) = f(\mathrm{A}(x))$$

is convex over the set $A^{-1}(\mathcal{S})$.

Proof. See [90, Section 7.4, Theorem 7.17]. □

For an extended list of convexity-preserving operations, we refer the reader to [89, 90, 92].

2.4.2.3 Subgradients

Since many important convex functions such as the absolute function are not continuously differentiable, we next introduce the subgradient. It generalizes the first-order condition of convex functions (2.7) to arbitrary tangent vectors as the following definition states.

Definition 2.4.8 (Subgradient) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be a proper function and let $x \in \operatorname{dom}(f)$. $g \in \mathbb{V}^*$ is called a* subgradient *of f at x if*

$$f(y) \geq f(x) + \langle g, y - x \rangle \tag{2.8}$$

holds for all $y \in \mathbb{V}$.

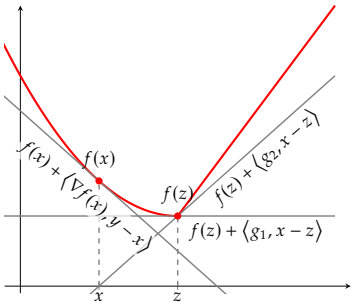


Figure 2.8: Illustration of subgradients.

Consequently, the subgradient is a linear functional in the dual space endowed with the dual norm $\|\cdot\|_*$. We use Riesz' representation theorem (Theorem 2.1.3) to identify $\mathbb{V}$ and its dual space $\mathbb{V}^*$, which implies that the duality pairing $\langle \cdot, \cdot \rangle$ can be identified with the usual scalar product of $\mathbb{V}$. Hence, the subgradient g can be uniquely identified by a vector in $\mathbb{V}$. However, its length is determined using the dual norm $\|\cdot\|_*$ of $\mathbb{V}^*$.

Geometrically a subgradient is defined as a linear tangent function that globally underestimates the function as visualized in Figure 2.8. At the point z we see that there are arbitrarily many possible subgradients, which are subsumed into the subdifferential.

Definition 2.4.9 (Subdifferential) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be a proper function and let $x \in \operatorname{dom}(f)$. The set of all subgradients of f at x is called the* subdifferential *of f at x and defined as*

$$\partial f(x) := \left\{ g \in \mathbb{V}^* : f(y) \geq f(x) + \langle g, y - x \rangle, \forall y \in \mathbb{V} \right\}.$$

Note that if f is differentiable at the point x, its subdifferential is uniquely determined by the gradient, i.e. $\partial f(x) = \{\nabla f(x)\}$. In fact, the subdifferential of a proper convex function is non-empty everywhere on its domain as the next theorem shows.

Theorem 2.4.16 (non-emptyness of Subdifferential of Convex Functions) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be a proper convex function and suppose $x \in \operatorname{int}(\operatorname{dom}(f))$. Then, $\partial f(x)$ is non-empty and closed.*

Proof. See [84, Section 3.2, Theorem 3.14]. □

This gives rise to a more general necessary and sufficient optimality condition for convex functions.

Theorem 2.4.17 (Optimality Condition for a Convex Function) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be a proper convex function. Then*

$$x^* \in \operatorname*{argmin}_{x \in \mathbb{V}} f(x)$$

if and only if $0 \in \partial f(x^*)$.

Proof. Due to subgradient inequality (2.8) we have

$$f(x) \geq f(x^*) + \left\langle g, x - x^* \right\rangle \overset{g=0 \in \partial f(x^*)}{=} f(x^*) \quad \forall x \in \mathbb{V}.$$

□

Thus, a point is a global minimum of a convex function if 0 is in its corresponding subdifferential.

2.4.2.4 Convex Conjugate

The convex conjugate also known as conjugate function or Fenchel conjugate is a core concept of convex analysis. The convex conjugate of a function defined on a vector space is a function on the associated dual space. Formally, the convex conjugate is defined as follows:

Definition 2.4.10 (Convex Conjugate) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be an extended real-valued function. The function $f^* : \mathbb{V}^* \to \overline{\mathbb{R}}$ defined as*

$$f^*(y) = \sup_{x \in \mathbb{V}} \left\langle y, x \right\rangle - f(x)$$

for any $y \in \mathbb{V}^*$ is called the convex conjugate.

As in the previous section, we use Riesz' representation theorem to identify the duality pairing with the scalar product of $\mathbb{V}$. This definition of convex conjugate holds for any extended real-valued functions including non-convex functions, indicator functions, and non-closed functions. Yet, all convex conjugate functions are convex and closed as stated in the following theorem.

Theorem 2.4.18 (Convexity and Closedness of Convex Conjugate) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be an extended real-valued function. Then the associated conjugate function f^* is closed and convex.*

Proof. See [84, Section 4.1, Theorem 4.3]. □

Another important property of convex conjugates is that they preserve the properness of convex functions.

Theorem 2.4.19 (Properness of the Convex Conjugate) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be a proper convex function. Then f^* is proper.*

Proof. See [84, Section 4.1, Theorem 4.5]. □

Frequently, we use the convex conjugate of a norm ($f(x) = \|x\|$), which is given by

$$f^*(y) = \delta_{\overline{\mathcal{B}}_{\|\cdot\|_*}(0,1)}(y) = \begin{cases} 0 & \text{if } \|y\|_* \leq 1 \\ \infty & \text{else} \end{cases}.$$

Hence, the convex conjugate of a norm is the indicator function of the closed unit dual norm ball [84, Section 4.4.12].

Applying the convex conjugate operation twice results in the so-called biconjugate, which is frequently used to derive dual optimization problems.

Definition 2.4.11 (Biconjugate) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be an extended real-valued function. The function $f^{**} : \mathbb{V} \to \overline{\mathbb{R}}$ defined as*

$$f^{**}(x) = \sup_{y \in \mathbb{V}^*} \langle x, y \rangle - f^*(y)$$

for any $y \in \mathbb{V}^*$ is called the biconjugate.

Recall that in our finite-dimensional setting we have $\mathbb{V}^{**} \cong \mathbb{V}$. The biconjugate of a function is convex and closed due to Theorem 2.4.18. Additionally, the biconjugate defines a lower bound on its associated function, as the next result proves.

Lemma 2.4.20 *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be an extended real-valued function. Then $f(x) \geq f^{**}(x)$ for all $x \in \mathbb{V}$.*

Proof. See [84, Section 4.2, Lemma 4.7]. □

A fundamental property of the biconjugate f^{**} is that if f is proper, closed, and convex, then the biconjugate is not just a lower bound but in fact equal to f.

Theorem 2.4.21 *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be a proper, closed, and convex function. Then $f^{**} = f$.*

Proof. See [84, Section 4.2, Theorem 4.8]. □

2.4.2.5 Proximal Operator

The proximal operator is used in many convex optimization methods to work with non-smooth functions. Before we can define the proximal operator, we need to introduce infimal convolutions.

Definition 2.4.12 (Infimal Convolution) *Let $f, g : \mathbb{V} \to \overline{\mathbb{R}}$ be two proper functions. The* infimal convolution *of f and g is given by*

$$(f \square g)(x) = \inf_{z \in \mathbb{V}} f(z) + g(x - z).$$

If the infimum is attained, then the infimal convolution is called exact.

The infimal convolution has the nice property that it preserves convexity under the conditions stated in the next theorem.

Theorem 2.4.22 (Convexity of Infimal Convolution) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be a proper convex function and let $g : \mathbb{V} \to \mathbb{R}$ be a real-valued function. Then $f \square g$ is convex.*

Proof. See [84, Section 2.3.2, Theorem 2.19]. □

A very important function in convex analysis is the *Moreau envelope*. For a proper convex and closed function f and $\lambda > 0$ the Moreau envelope is defined as the infimal convolution of f with the squared norm scaled by λ, i.e.

$$f_\lambda(x) = \left(f \square \tfrac{1}{2\lambda} \| \cdot \|^2 \right)(x).$$

The Moreau envelope has a smoothing effect and λ is called the smoothing parameter [84, Section 6.7]. Figure 2.9 visualizes this smoothing effect of the Moreau envelope for the absolute function $f(x) = |x|$. The resulting smooth function is also known as the Huber function [37] in robust statistics.

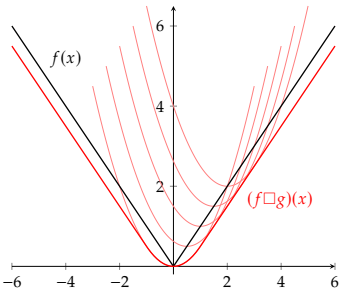


Figure 2.9: Illustration of the Moreau envelope of $f(x) = |x|$ with $\lambda = 1$.

The proximal operator of a function f is then simply defined as the minimizing argument of the corresponding Moreau envelope.

Definition 2.4.13 (Proximal Operator) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be a proper, closed and convex function and $\lambda > 0$. The* proximal operator *of f is given by*

$$\operatorname{prox}_{\lambda f}(x) := \operatorname*{argmin}_{z \in \mathbb{V}} f(z) + \tfrac{1}{2\lambda} \|z - x\|^2 .$$

for any $x \in \mathbb{V}$.

We will frequently use the abbreviation prox to denote the proximal operator. Coming back to the previous example $f(x) = |x|$, the proximal operator of f is given by the soft shrinkage operator

$$\operatorname{prox}_{\lambda|\cdot|}(x) = \max(|x| - \lambda, 0)\operatorname{sgn}(x),$$

which is illustrated in Figure 2.10.

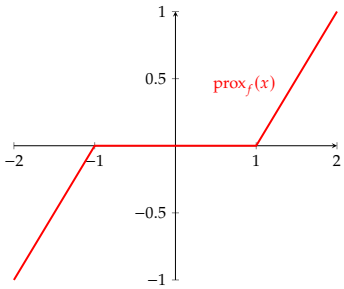


Figure 2.10: Proximal operator of $f(x) = |x|$ with $\lambda = 1$.

The next theorem ensures existence and uniqueness of the proximal operator.

Theorem 2.4.23 (Existence and Uniqueness of Prox) *Let $f : \mathbb{V} \to \overline{\mathbb{R}}$ be a proper closed and convex function. Then, $\operatorname{prox}_f(x)$ exists and is unique for any $x \in \mathbb{V}$.*

Proof. See [84, Section 6.2, Theorem 6.3]. □

In optimization, we are frequently minimizing a function $f : \mathbb{V} \to \mathbb{R}$ over a non-empty closed and convex constraint set $\mathcal{S} \subset \mathbb{V}$. One way to account for the constraint is to incorporate its indicator function $\delta_{\mathcal{S}} : \mathbb{V} \to \overline{\mathbb{R}}$, which is a proper closed and convex function. Applying the proximal operator to an indicator function is equivalent to applying the orthogonal projection [84, Section 6.4], i.e.

$$\operatorname{prox}_{\delta_{\mathcal{S}}}(x) = \operatorname{proj}_{\mathcal{S}}(x)$$

for all $x \in \mathbb{V}$, where the *orthogonal projection operator* is defined as

$$\operatorname{proj}_{\mathcal{S}}(x) = \operatorname*{argmin}_{z \in \mathcal{S}} \tfrac{1}{2} \|z - x\|^2 .$$

The next result shows that the existence theorem for the prox operator can be extended to the orthogonal projection operator.

Theorem 2.4.24 (Existence and Uniqueness of Projection) *Let $\mathbb{V}$ be a Hilbert space and let $\mathcal{S} \subset \mathbb{V}$ be a non-empty, closed and convex set. Then $\operatorname{proj}_{\mathcal{S}}(x)$ exists and has a unique optimal solution for every $x \in \mathbb{V}$.*

Proof. See [90, Section 8.3, Theorem 8.8]. □

This unique solution is characterized by the shortest distance to the set as depicted in Figure 2.11.

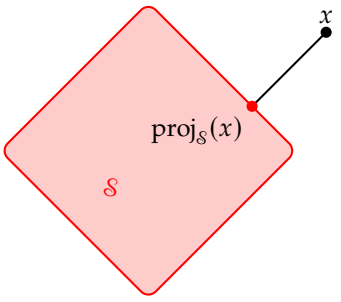


Figure 2.11: Visualization of the orthogonal projection of a point x onto the convex set $\mathcal{S}$.

2.4.3 First-order Methods

In this section, we elaborate on the efficient computation of solutions of the optimization problem (2.6) and discuss suitable algorithms. In the following, we assume that a minimizer of (2.6) exists, which could be shown, for instance, by Theorem 2.1.6 or Theorem 2.1.7. Thus, we consider minimization problems of the form

$$\min_{x \in \mathcal{S}} f(x),$$

where $f : \mathcal{S} \to \overline{\mathbb{R}}$ is an extended real-valued function over a convex set $\mathcal{S} \subset \mathbb{R}^n$. In fact, we can drop the constraint $x \in \mathcal{S}$ by adding the indicator function $\delta_{\mathcal{S}}$ to the objective function, as we have discussed in the previous section. Then, we end up with the unconstrained problem

$$\min_{x \in \mathbb{R}^n} f(x) \tag{2.9}$$

for an extended real-valued function $f : \mathbb{R}^n \to \overline{\mathbb{R}}$. Here, we focus on first-order optimization methods that generate a sequence $x^k \in \mathbb{R}^n$ incorporating only gradient information. The use of second-order methods such as Newton's method [90, Section 5] is often not possible in typical imaging and machine learning applications due to memory and time constraints.

Nemirovsky and Yudin [94] developed lower complexity bounds for any first-order method for smooth convex objective functions. They showed that no first-order method can in general converge faster than $\mathcal{O}\left(\frac{1}{k^2}\right)$ [1]. On the other hand, if f is convex and non-differentiable, then the lower complexity bound is $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ due to Nesterov [92, 95].

For non-convex objective functions, we do not have these complexity bounds for first-order methods. Moreover, we can only ensure converge to a stationary point $x \in \mathbb{R}^n$ characterized by the first-order condition $\nabla f(x) = 0$, which can be a local/global minimum/maximum or a saddle point. Therefore, the convergence rate is typically stated w.r.t. the first-order condition[2]. In this general non-convex setting, the best proven convergence rate w.r.t. the first-order condition is $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ [91, 96, 97], which is actually a rather bad rate.

1: In the convex case, all rates are w.r.t. the function values if not stated otherwise. For example, a rate $\mathcal{O}\left(\frac{1}{k^2}\right)$ implies that

$$f(x^k) - f^* \leq \mathcal{O}\left(\frac{1}{k^2}\right),$$

where $f(x^k)$ is the objective function value of the k^{th} iterate (sequence element) and f^* is the objective function value of the optimal solution. This means that the difference of the objective function values decrease proportionally to $\frac{1}{k^2}$ if k iteration steps are performed.

2: In the non-convex case, convergence rates are w.r.t. the first-order characterization. For example, a rate $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ ensures that

$$\left\|\nabla f(x^k)\right\| \leq \mathcal{O}\left(\frac{1}{\sqrt{k}}\right).$$

This means that the norm of the gradient of the k^{th} iteration is proportional to $\frac{1}{\sqrt{k}}$.

2.4.3.1 Gradient Methods

Gradient methods assume that the objective function f of problem (2.6) is continuously differentiable, i.e. $f \in C^1(\mathbb{R}^n, \mathbb{R})$. The simplest gradient method is gradient descent, which generates a sequence of points x^k by following the negative gradient of the objective function starting from an initial point $x^0 \in \mathbb{R}^n$. Hence, in every iteration step k it follows the steepest descent direction, as listed in Algorithm 1. The step

Algorithm 1: Gradient Descent

Initialization: Set $k = 0$, choose a starting point $x^0 \in \mathbb{R}^n$ and a step size sequence τ^k.

1 **while** *not converged* **do**
2 $\quad x^{k+1} = x^k - \tau^k \nabla f(x^k)$
3 $\quad k = k + 1$

size τ^k can be selected in various ways and we refer the reader for an overview of suitable step size selection methods to [90, Section 4.2]. Here, we assume that the objective function has a Lipschitz continuous gradient with Lipschitz constant L, as specified in Definition 2.1.8. Then, the gradient method converges for constant step

sizes $\tau^k = \tau \in \left(0, \frac{2}{L}\right)$ [91]. If the objective function f is convex, one can show that the function values converge with rate $\mathcal{O}\left(\frac{1}{k}\right)$ to the optimum [91]. In contrast, if the objective function f is non-convex, one can show that gradient descent with constant step size converges with rate $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ to a stationary point. Thus, gradient descent is not an optimal method in the convex case since its convergence rate is an order of magnitude slower than the optimal lower bound.

To close the gap to the optimal convergence rate for smooth convex objective functions, Nesterov [98] proposed to extend gradient descent by incorporating an extrapolation step in a smart way. The resulting algorithm is nowadays known as Nesterov's accelerated gradient method and is listed in Algorithm 2. Compared to gradient

Algorithm 2: Nesterov's Accelerated Gradient Method

Initialization: Set $k = 0$, $\alpha^0 = 1$, choose a starting point $x_0 \in \mathbb{R}^n$ and a step size sequence τ^k.

1 $x^{-1} = x^0$
2 **while** *not converged* **do**
3 $\quad \alpha^{k+1} = \frac{1+\sqrt{1+4(\alpha^k)^2}}{2}$
4 $\quad \beta^k = \frac{\alpha^k - 1}{\alpha^{k+1}}$
5 $\quad \widetilde{x}^k = x^k + \beta^k \left(x^k - x^{k-1}\right)$
6 $\quad x^{k+1} = \widetilde{x}^k - \tau^k \nabla f(\widetilde{x}^k)$
7 $\quad k = k + 1$

descent, this algorithm only requires additional memory to store the previous iteration point x^{k-1} and has almost no computational overhead. The step size τ^k can be selected by similar considerations as before. If the objective function has a Lipschitz continuous gradient with Lipschitz constant L, Nesterov's method converges for $\tau^k = \tau \in \left(0, \frac{1}{L}\right]$. Nesterov showed in his seminal paper [98] that this method converges to the optimal function value with the rate $\mathcal{O}\left(\frac{1}{k^2}\right)$ for smooth convex objective functions. Recently, Ghadimi and Lan [97] published a proof for smooth non-convex objective functions and showed that Nesterov's accelerated gradient method converges to a stationary point with the rate $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$, which is the same rate as gradient descent.

2.4.3.2 Proximal Methods

The urge to optimize non-differentiable objective functions led to the development of proximal methods. The majority of proximal methods assume that the objective function can be decomposed into a continuously differentiable function $g \in C^1(\mathbb{R}^n, \mathbb{R})$ with Lipschitz continuous gradient and a proper and closed function $h : \mathbb{R}^n \to \overline{\mathbb{R}}$. Then, the corresponding minimization problem reads as

$$\min_{x \in \mathbb{R}^n} \left\{ f(x) = g(x) + h(x) \right\}. \tag{2.10}$$

In fact, many applications in computer vision, image reconstruction, and medical imaging can be cast into this form [10]. We denote by L the Lipschitz constant of ∇g.

If both g and h are convex functions, Beck and Teboulle [99] proposed an algorithm called fast iterative shrinkage and thresholding algorithm (FISTA) that converges to the optimal function with the optimal rate $\mathcal{O}\left(\frac{1}{k^2}\right)$. They extended Nesterov's accelerated gradient method (Algorithm 2) by performing a proximal gradient step on the composite function, see line 6 of Algorithm 3. The term shrinkage and thresholding in the algorithm's name originates from the soft shrinkage operator, which is the proximal operator associated with the absolute function, see Section 2.4.2.5. In their

Algorithm 3: Fast Iterative Shrinkage and Thresholding Algorithm

Initialization: Set $k = 0$, $\alpha^0 = 1$, choose a starting point $x_0 \in \mathbb{R}^n$.

$x^{-1} = x^0$
while *not converged* **do**
  $\alpha^{k+1} = \frac{1+\sqrt{1+4(\alpha^k)^2}}{2}$
  $\beta^k = \frac{\alpha^k - 1}{\alpha^{k+1}}$
  $\widetilde{x}^k = x^k + \beta^k \left(x^k - x^{k-1}\right)$
  $x^{k+1} = \operatorname{prox}_{\frac{1}{L}h}\left(\widetilde{x}^k - \frac{1}{L}\nabla g(\widetilde{x}^k)\right)$
  $k = k + 1$

paper [97], Ghadimi and Lan also considered this composite problem and analyzed its convergence rate for g being non-convex and reported refined rates.

2.4.3.3 Primal-dual Hybrid Gradient Method

In the previous section, at least one function of the composite problem (2.10) was continuously differentiable. Here, we consider the composite problem

$$\min_{x \in \mathbb{R}^n} f(Kx) + g(x), \tag{2.11}$$

where $K \in \mathbb{R}^{m \times n}$ is the matrix representation of a linear operator and both $f : \mathbb{R}^m \to \overline{\mathbb{R}}$ and $g : \mathbb{R}^n \to \overline{\mathbb{R}}$ are proper, closed and convex functions. Thus, both f and g might be non-differentiable. Since f is proper, closed and convex, we have $f^{**} = f$, see Theorem 2.4.21. Hence, we can rewrite (2.11) as

$$\min_{x \in \mathbb{R}^n} \max_{y \in \mathbb{R}^m} \left\langle Kx, y \right\rangle - f^*(y) + g(x), \tag{2.12}$$

where $f^* : \mathbb{R}^m \to \overline{\mathbb{R}}$ is the convex conjugate of f (Definition 2.4.10). Note that we used Riesz' representation theorem (Theorem 2.1.3) to identify the duality pairing with the usual dot product on $\mathbb{R}^m$. This saddle point formulation of (2.11) is called the corresponding primal-dual problem since the objective function is a problem of the primal variable x and the dual variable y. Chambolle and Pock [66] proposed the primal-dual hybrid gradient algorithm to efficiently find a saddle point of (2.12). The underlying idea is to alternate between a proximal gradient descent step on the primal variable and a proximal gradient ascent step on the dual variable using the extrapolated primal variable, see Algorithm 4. This algorithm converges if the step

Algorithm 4: Primal-dual Hybrid Gradient Algorithm

Initialization: Set $k = 0$, choose starting points $x_0 \in \mathbb{R}^n$, $y_0 \in \mathbb{R}^m$ and step sizes $\tau, \sigma > 0$.

while *not converged* **do**
  $x^{k+1} = \operatorname{prox}_{\tau g}\left(x^k - \tau K^* y^k\right)$
  $y^{k+1} = \operatorname{prox}_{\sigma f^*}\left(y^k + \sigma K\left(2x^{k+1} - x^k\right)\right)$
  $k = k + 1$

sizes are chosen such that

$$\tau\sigma \left\|K\right\|^2 < 1,$$

where $\|K\|$ is the induced norm of the matrix K. In this case, Algorithm 4 converges with rate $\mathcal{O}\left(\frac{1}{k}\right)$, which is optimal for non-smooth and convex problems [95].

The primal-dual hybrid gradient algorithm is widely used in convex optimization due to its simplicity and flexibility. In recent years, it has been extended to support

nonlinear operators [100] and stochastic sampling of operators [101]. In addition, Malitsky and Pock [102] suggested to replace the fixed step size selection by a line search alternative.

2.4.3.4 Methods for Additive Cost Problems

A ubiquitous optimization problem in machine learning is the minimization of an additive cost function w.r.t. model parameters. Therefore, we consider in this section optimization problems of the form

$$\min_{x \in \mathcal{S}} \sum_{i=1}^{m} f_i(x), \tag{2.13}$$

where $f_i : \mathbb{R}^n \to \mathbb{R}$ for $i = 1, \ldots, m$ are component functions and $\mathcal{S} \subset \mathbb{R}^n$ is a convex set. Here, the objective function is given by the sum of all component functions assigning a cost to each sample for given parameters x. Note that in this formulation the data samples are included in the component functions and are thereby 'hidden' in the notation. We assume that m is very large, which is true for many modern data sets in computer vision [103, 104], and medical imaging [13, 78]. In the remainder of this section, we introduce incremental gradient and incremental proximal gradient methods and discuss their convergence for convex and non-convex component functions. We conclude this section by discussing the Adam optimizer, which can be interpreted as an incremental method and is one of the most widespread optimization algorithms in deep learning.

A class of optimization algorithms especially designed for additive cost problems of the form (2.13) are incremental gradient methods [105, 106] and incremental proximal methods [81, 107]. While incremental gradient methods assume that each component function is differentiable, incremental proximal methods also allow non-differentiable component functions. The basic idea of incremental methods is to operate on a single component function f_c in each minimization step in order to speed up the optimization procedure.

The basic incremental gradient method is also known as online backpropagation [108] and has the form

$$x^{k+1} = \operatorname{proj}_{\mathcal{S}} \left(x^k - \tau^k \nabla f_{c(k)}(x^k) \right),$$

where τ^k defines the step size at iteration k and $c(k) : \mathbb{N} \to 1, \ldots, m$ selects the component for the k^{th} iteration. The basic differences between variants of this method are the selection of the step size τ^k and how the components are distributed to each iteration $c(k)$, which can be either random or deterministic. The convergence of all these variants was proven under various conditions. For instance, Mangasarian and Solodov [108] proved convergence for diminishing step sizes if $c(k)$ defines a cyclic order.

To account for non-differentiable component functions, Nedić and Bertsekas [81, 109] introduced incremental subgradient and proximal methods. The intuition behind incremental proximal methods is that the component functions can be partitioned in continuously differentiable functions $f_i \in C^1(\mathbb{R}^n, \mathbb{R})$ and proper, closed and convex functions $h_i : \mathbb{R}^n \to \overline{\mathbb{R}}$ for $i = 1, \ldots, m$ such that problem (2.13) changes to

$$\min_{x \in \mathcal{S}} \sum_{i=1}^{m} f_i(x) + h_i(x). \tag{2.14}$$

Then, each step of the incremental proximal gradient method [81] is of the form

$$x^{k+1} = \operatorname{prox}_{\tau^k h_{c(k)}} \left(x^k - \tau^k \nabla f_{c(k)}(x^k) \right),$$

where $\text{prox}_{\tau^k h_{c(k)}}$ is the proximal step on a single component $h_{c(k)}$ and $\nabla f_{c(k)}$ is the gradient of a single component selected by $c(k)$. τ^k denotes the step size of the k^{th} iteration step. In the case that all functions f_i for $i = 1, \dots, m$ of problem (2.14) are convex, Bertsekas [81] showed that the corresponding incremental proximal gradient method converges for cyclic and random component selection functions $c(k)$ using a diminishing step size provided that all functions are Lipschitz continuous and have a bounded gradient or subgradient, respectively. In addition, under the same assumption he proved approximate convergence in the function value, i.e. the sequence of the objective function converges to a ball around the optimal value, for a fixed suitably small step size $\tau^k = \tau > 0$ [81]. Later Sra [107] showed approximate convergence of the incremental proximal gradient method without assuming convexity of the differentiable component functions $f_i \in C^1(\mathbb{R}^n, \mathbb{R})$ for $i = 1, \dots, m$. In this proof, he considered a single non differentiable function $h : \mathbb{R}^n \to \overline{\mathbb{R}}$ and assumed bounded gradients or subgradients, respectively. Then, the incremental proximal gradient method also approximately converges in the function values for a cyclic component selection function $c(k)$ and a fixed step size $\tau^k = \tau > 0$.

Incremental gradient methods that select the component function randomly in each step are also known as stochastic gradient methods. Probably the most famous and widespread algorithm of this type of incremental methods is Adam (short for adaptive moment estimation) by Kingma and Ba [110]. The Adam algorithm combines incremental methods with the heavy ball method of Polyak [111] and a clever adaptive preconditioning of the gradient, see Algorithm 5. Adam computes running statistics

Algorithm 5: Adam

Initialization: Set $k = 0$, $m_0 = 0 \in \mathbb{R}^n$, and $v_0 = 0 \in \mathbb{R}^n$. Choose a starting point $x_0 \in \mathbb{R}^n$, $\epsilon > 0$, step size $\tau > 0$ and momentum parameters $\beta_1, \beta_2 \in [0, 1)$.

Default: $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and $\tau = 10^{-3}$

while *not converged* **do**

$g^k = \nabla f_{c(k)}(x^k)$ *(gradient of random component function)*

$m^k = \beta_1 m^{k-1} + (1 - \beta_1) g^k$ *(update first moment)*

$v^k = \beta_2 v^{k-1} + (1 - \beta_2)(g^k \odot g^k)$ *(update second moment)*

$\widehat{m}^k = \frac{m^k}{1-\beta_1^k}$ *(bias correction of first moment)*

$\widehat{v}^k = \frac{v^k}{1-\beta_2^k}$ *(bias correction of second moment)*

$x^{k+1} = x^k - \tau \frac{\widehat{m}^k}{\sqrt{\widehat{v}^k} + \epsilon}$

$k = k + 1$

of the first and second moment of the gradient and updates the iterate x^k using the first moment scaled by the second moment, which is a form of step size annealing. In addition, Adam incorporates a bias correction step for each moment to account for wrong initial estimates. In the last years, Adam has empirically proven that it works very well in many machine learning problems. In fact, Adam is the standard algorithm for training neural networks today although it may not converge [112].

3 Machine Learning

Starting from the invention of the metal oxide semiconductor field-effect transistor (MOSFET) in 1947 till up to 2018 it has been estimated that more than $13 \cdot 10^{21}$ MOSFETs have been produced[1]. This makes the MOSFET the most frequently manufactured human artifact in history. Moreover, by far more MOSFETs have been produced than there are stars in our galaxy[2]. Why? MOSFETs are the essential building blocks of computer chips, which we require to process today's never-ending stream of data. In 2020 every day $220 \cdot 10^9$ emails are sent, 4PB (petabyte — 10^{15} bytes) of data is generated on Facebook, including $350 \cdot 10^6$ photos and 10^5 hours of video, $5 \cdot 10^9$ search requests are posted and 28PB of data is generated from wearable devices such as smart watches[3]. To analyze and extract meaningful information from this flood of data, automated processing is inevitable. *Machine Learning* provides the necessary concepts and tools. It is the study of data-driven methods that are capable of detecting patterns in data and then using these extracted patterns to make predictions or decisions on unseen data [34, 113].

1 : David Laws, *13 Sextillion & Counting: The Long & Winding Road to the Most Frequently Manufactured Human Artifact in History*, Computer history museum, `https://bit.ly/324cf68`, accessed September, 2020.

2 : According to ESA there are something like 10^{11} to 10^{12} stars in our galaxy. `https://bit.ly/3lWbwvy`, accessed September 2020.

3 : Raconteur *A Day in Data*, `http://rcnt.eu/un8bg`, accessed September 2020.

In this chapter, we provide a brief introduction to machine learning. We first describe the different types of machine learning and elaborate on the difference between generative and discriminative learning. Then, we define neural networks (NNs) and their building blocks. Finally, we review network architectures and design patterns for NNs. For an in-depth analysis of the broad field of machine learning, we refer to the excellent text books [32, 34, 113–116].

3.1 Machine Learning Types

From a classical point of view, machine learning approaches can be classified into two main types [32, 34]. In *supervised learning* we are given a dataset consisting of input/output pairs to learn a parametric mapping from the inputs to the outputs. In contrast, in *unsupervised learning* we have a dataset consisting only of inputs and the task is to detect and extract underlying patterns of the data. A third style of machine learning that received more attention recently is *reinforcement learning* [116]. There, the task is to determine a policy of an agent's actions that manipulate a given environment such that an occasional reward is maximized. In the following sections, we discuss supervised/unsupervised learning and elaborate on the difference between generative and discriminative learning approaches. We refer the interested reader to [116] for more details on reinforcement learning.

3.1.1 Supervised Learning

Supervised learning is by far the most widespread machine learning approach [34, 115]. Here, we develop supervised learning from a probabilistic point of view. To model the sampling of data points, we consider a complete probability space $(\Omega, \mathfrak{F}, \mathbb{P})$, where Ω denotes the sampling space, $\mathfrak{F}$ the corresponding σ-algebra and $\mathbb{P}$ the probability measure. The data points are generated by means of an input random variable $X : \Omega \to \mathcal{X}$ and an output random variable $Y : \Omega \to \mathcal{Y}$, where $\mathcal{X}$ is the input space and $\mathcal{Y}$ the output space. Both random variables follow the joint distribution $(X, Y) \sim \mathcal{T}$ and a realization consists of an *input* $x \in \mathcal{X}$ and a target *output* $y \in \mathcal{Y}$. The input x describes essential features of the considered data. In the case of spam detection, for instance, the input would be encoded features of an email

including the message body and metadata, while in the case of image processing the input is typically given by the intensity values of a discrete image. Thus, the input space can frequently be modeled as a subset of the space of n-dimensional feature vectors $\mathcal{X} \subset \mathbb{R}^n$. In contrast, the output space $\mathcal{Y}$ is usually defined by the task. For the spam detection example, the output space would be a binary set $\mathcal{Y} = \{0, 1\}$ where 1 indicates that a message is spam, while 0 indicates that it is not. On the other hand, the target output y for image processing is often of the same size as the input space, e.g. image denoising or image deblurring. Based on the output space $\mathcal{Y}$ we can separate supervised learning methods into classification ($|\mathcal{Y}| < \infty$) and regression problems ($|\mathcal{Y}| = \infty$).

In either case, the aim is to learn a parametric mapping

$$f : \mathcal{X} \times \Theta \to \mathcal{Y}$$

called the *model* that transforms each input $x \in \mathcal{X}$ into an output $f(x, \theta) \in \mathcal{Y}$ given the *parameters* of the model $\theta \in \Theta$. The quality of a model is determined by a *cost* function $\mathrm{J} : \Theta \to \mathbb{R}$ that assigns to each parameter value $\theta \in \Theta$ a real-valued cost decreasing with increasing quality. The cost function is frequently of the form

$$\mathrm{J}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{T}} \left[\ell(f(x, \theta), y) \right], \tag{3.1}$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is the loss function. We frequently call this cost function the *expected loss*. Then, we call the process of determining the parameters θ such that the cost function J is minimal *training* or *learning*. In summary, in supervised learning, we train the parameters of the model such that its output is close to the desired target output according to the loss function for each sample of the data distribution.

3.1.1.1 Classification

Classification is the process of detecting features of objects that enable a categorization into a set of classes [34]. To account for different classes, the output space is typically a set of class labels $\mathcal{C} = \{1, \ldots, C\}$, where C denotes the number of possible classes. If a task only considers two mutually exclusive classes ($C = 2$), it is called a *binary* classification task. A simple binary classification task is the categorization of chess pieces based on their color, which is either white or black ($C = 2$). On the other hand, a task with multiple mutually exclusive classes ($C > 2$) is called *multiclass* classification. If we extend our chess classification problem from before to also account for the type[1] of a chess piece, we end up with a multiclass classification problem including $C = 12$ mutually exclusive classes.

1: There are six types of chess pieces: pawn, rook, knight, bishop, queen, and king

Often we as humans have problems telling objects apart. For example, if we look at a picture of a single chess piece colored in middle brown shades, we cannot be sure whether it is actually white or black. To remove any doubt, we would need the color information of the opponent player. Thus, it is reasonable to equip a classification model with the possibility to express its uncertainty. This can be done by relaxing the output space of the model to the unit simplex $\mathcal{Y} = \Delta_C := \left\{ y \in \mathbb{R}^C : \sum_{i=1}^{C} y_i = 1, y_i \geq 0 \text{ for } i \in \mathcal{C} \right\}$. Then, the i^{th} entry of each output of a classification model can be interpreted as the discrete conditional probability density $y_i = p(c = i|x)$. Using this probabilistic output space, we can use the model output $\widehat{y}_i := f(x, \theta) \in \Delta_C$ to determine the class label of the most favored class by computing

$$\widehat{c} \in \operatorname*{argmax}_{i \in \mathcal{C}} \widehat{y}_i$$

for a certain input $x \in \mathcal{X}$ and model parameters $\theta \in \Theta$, which yields the optimal decision under the zero-one loss [113]. In addition, we can define the target output in this representation. Let $c \in \mathcal{C}$ denote the target class label. Then, the corresponding probabilistic representation is given by $y \in \Delta_C$ such that $y_c = 1$ and $y_i = 0$ for

$i \in \mathcal{C} \setminus \{x\}$. Probably the most used loss function for this probabilistic representation of the classification problem is the *cross entropy* $\mathrm{h} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. It is defined as

$$\ell(y, \widehat{y}) = \mathrm{h}(y, \widehat{y}) \coloneqq -\sum_{i=1}^{C} y_i \log(\widehat{y}_i).$$

The cross entropy is a robust loss function that exponentially increases along with the distance to the target label, as illustrated in Figure 3.1.

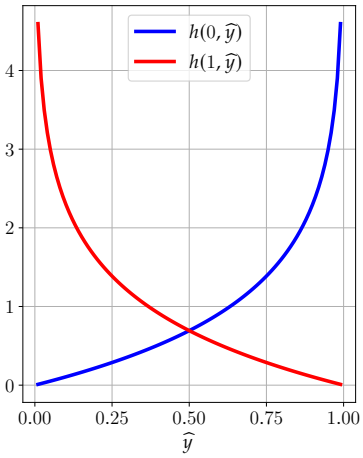


Figure 3.1: Illustration of the cross entropy loss function for a binary classification problem $C = 2$. The blue curve represents the loss if the target class label is 0 and the red curve the loss if the target class label is 1, respectively.

As a first classification example we consider the classification of the so-called moons dataset, depicted in Figure 3.2. This dataset consists of samples drawn from two arcs that are intertwined in a two-dimensional plane. A blue dot indicates that a sample stems from the first class, while a red cross marks samples from the second class. Since $\mathcal{Y} = \Delta_2 = \{(y_1, y_2) \in \mathbb{R}^2 : y_1, y_2 \geq 0, y_1 + y_2 = 1\}$, we can simplify the output space for this binary classification task to $\mathcal{Y} = [0, 1]$. Then, the output $\widehat{y} \coloneqq f(x, w) \in [0, 1]$ of the model represents the probability that a sample corresponds to class 1. The associated probability that the output corresponds to class 2 is then simply given by $1 - f(x, w)$. In this case, the cross entropy loss changes to

$$\mathrm{h}(y, \widehat{y}) = -y \log(\widehat{y}) - (1 - y) \log(1 - \widehat{y}),$$

which is known as the *binary cross entropy* loss. We used this loss to learn the parameters θ of a simple model, also called a *classifier* in this context, and visualized its output for the entire 2-dimensional plane in the background of Figure 3.2. The model output is color-coded such that 0 is represented by the blue color, 0.5 by the white, and 1 by the red color. The learned classifier (actually the learned parameters θ along with the model f) splits the entire plane into two distinct regions in the upper left and lower right corner. Between these two regions is a transition area, which indicates the uncertainty of the classifier.

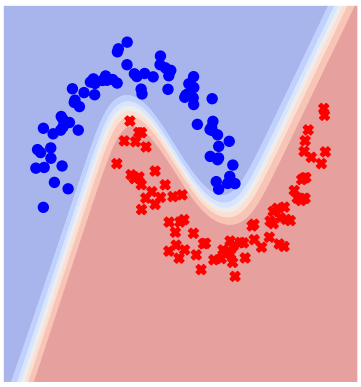


Figure 3.2: Visualization of a binary classification task. The blue dots indicate the first class, while the red crosses mark samples of the second class. The space is classified into two regions illustrated by the background, where the blue region is associated with class one and the red region with class two, respectively. The decision boundary between these areas is indicated by the bright area.

As a second example, we consider the task of semantic image segmentation of hematoxylin and eosin (H&E) stained tissue sections, which are typically used in pathology to detect various types of cancer [117]. In general, the task of semantic segmentation is defined as assigning to each pixel of an image a semantic class label. In the case of tissue sections, these labels are associated with different cell types. Figure 3.3 depicts a sample image from the MONUSAC-2020 challenge [118], where four different cell types are highlighted. Epithelial cells are shown in red, lymphocytes in yellow, macrophages in green, and neutrophils in blue. In this particular task, we have to assign to each pixel a class label in $\{1, \ldots, 5\}$, where the fifth label indicates that the pixel corresponds to neither of the previously mentioned cell types. Typically, a H&E stained image of size $n_1 \times n_2$ with 3 color channels is represented by a vector $x \in \mathcal{X} = \mathbb{R}^n$ with $n = 3n_1n_2$. Then, a semantic segmentation model predicts for each pixel a discrete probability distribution over all class labels. The corresponding output space is given by

$$\mathcal{Y} = \Delta_C^n \coloneqq \left\{ y \in \mathbb{R}^{n \times C} : \sum_{j=1}^{C} y_{i,j} = 1, y_{i,j} \geq 0 \text{ for } i = 1, \ldots, n,\ j \in \{1, \ldots, C\} \right\},$$

where the number of possible classes is $C = 5$ in this particular case. Since every pixel needs to be classified, the cost function must account for all pixels. A typical choice in semantic segmentation is to compute the cross entropy over all pixels. Hence, the aggregated cost function w.r.t. the model parameters θ reads as

$$\mathrm{J}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{T}} \left[\sum_{i=1}^{n} \mathrm{h}(f(x, \theta)_i, y_i) \right]$$

where $f(x, \theta)$ denotes the model output and y the target output both encoded in the probabilistic output representation. The subscript i indicates that the cross entropy

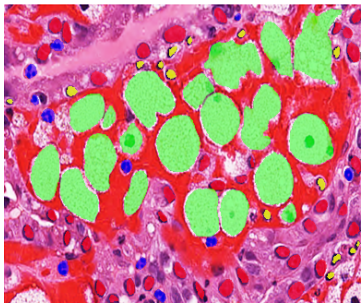


Figure 3.3: Visualization of a sample from the MONUSAC-2020 dataset (`https://monusac-2020.grand-challenge.org`, accessed September, 2020.) Here, the task is to detect and segment different types of cells indicated by the green, red, blue, and yellow regions.

is computed for the discrete probability distribution over the class labels of the i^{th} pixel.

3.1.1.2 Regression

As we have seen in the previous section, many problems in real-world applications are classification problems. In this section, we address the question: What happens when the number of possible output classes of a task is infinite? In machine learning, these tasks are called *regression* problems and are characterized by continuous output spaces [32, 34]. They occur in various application fields, for instance, in finance we would like to predict the stock price of certain shares given their history. Another example from engineering is that we would like to mimic the dynamics of a physical process by a parametric model that can be easily evaluated [119]. Also in medicine regression problems appear frequently, for instance, consider the tasks of accelerated magnetic resonance imaging (MRI) reconstruction [120] and low-dose computed tomography (CT) reconstruction [67], where we would like to estimate an entire image given measurements in the k-space or sinogram.

Clearly, for regression problems, it is challenging to infer a nonparametric probability distribution over all possible outputs as before due to limited memory. Therefore, we neglect the probabilistic representation of the model output and directly predict the desired output, which is typically a real-valued vector $y \in \mathcal{Y} \subset \mathbb{R}^m$. Thus, the output of regression models is in many cases a point estimate in a continuous vector space.

Let us consider the simple one-dimensional regression problem depicted in Figure 3.4. The blue dots indicate measurements $z = y(x) + \xi$, where $y : \mathbb{R} \to \mathbb{R}$ is the functional dependency of the input $x \in \mathbb{R}$ that we would like to estimate, and $\xi \in \mathbb{R}$ is additive Gaussian noise with unknown variance σ^2, i.e. $\xi \sim \mathcal{N}(0, \sigma^2)$. To approximate the function y, we use the class of polynomials of degree d. In detail, the model output for an input $x \in \mathbb{R}$ is computed by

$$f(x, \theta) = \sum_{i=0}^{d} \theta_i x^i = \begin{pmatrix} 1 & x & x^2 & \cdots & x^d \end{pmatrix} \theta, \tag{3.2}$$

where $\theta \in \mathbb{R}^{d+1}$ are the parameters of the polynomial. Assume that we have $N \in \mathbb{N}$ sample pairs $(x_i, z_i)_{i=1}^N$. Then, the expected least squares loss over these samples reads as

$$\mathrm{J}(\theta) = \frac{1}{N} \sum_{i=1}^{N} (f(x_i, \theta) - z_i)^2 = \|X\theta - z\|_2^2,$$

where the feature matrix $X \in \mathbb{R}^{N \times d+1}$ is given by

$$X = \begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^d \\ 1 & x_2 & x_2^2 & \cdots & x_2^d \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^d \end{pmatrix} \tag{3.3}$$

and the measurement vector $z = (z_1 \ \ldots \ z_N)^\top \in \mathbb{R}^N$ represents all individual measurements. Since J is a proper, smooth and convex function, its global minimum is given by setting its gradient w.r.t. the parameters θ to zero, which results in

$$\theta = (X^\top X)^{-1} X^\top z. \tag{3.4}$$

The thereby obtained polynomial of degree $d = 3$ is depicted as the red curve in Figure 3.4. It describes the data samples well and also seems to correctly characterize the underlying functional dependency y.

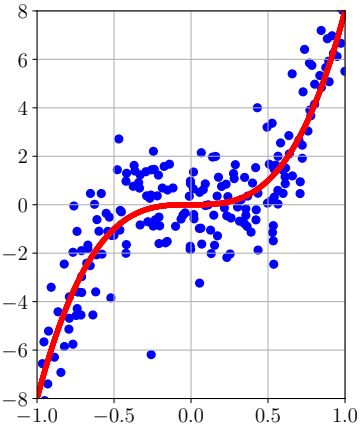


Figure 3.4: Illustration of a regression task. Given data samples (blue dots), the task is to fit the parameters θ of a polynomial (red line) such that it best explains the data in the least squares sense.

Another regression task example from computer vision is the estimation of optical flow [121]. The optical flow between two images of an image sequence is defined as the apparent motion of pixels. Figure 3.5 depicts a sample from the Sintel dataset [122]. The optical flow (bottom) specifies for each pixel in the reference image (top) the relative position to the corresponding pixel in the second image (middle). As a result, the optical flow computes the linear motion (direction and speed) of pixels. Since objects move freely and continuously in the real work, the optical flow is not necessarily aligned with the pixel grid and thus is a prototypical regression task. Due to the huge diversity of image sequences and the endless number of motion patterns, computing the optical flow is a challenging nonlinear inverse problem. Hence, there is no hope that such a simple linear model as in the previous example performs well for arbitrary scenes. Therefore, a broad variety of parametric approaches has evolved [123–125] in recent years.

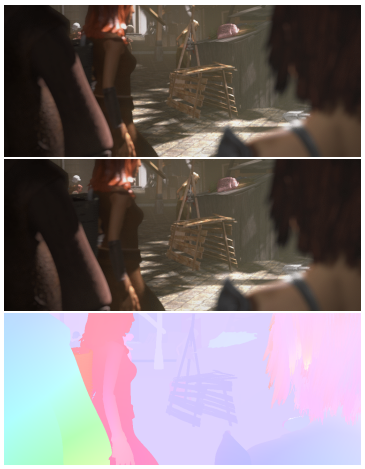


Figure 3.5: The optical flow represents the motion of each pixel from the reference image (top) to the target image (middle). It is depicted at the bottom image using a color code in the HSV color space, where the hue specifies the motion direction and the value the motion length of each pixel. This is a sample from the Sintel dataset [122].

3.1.2 Unsupervised Learning

In unsupervised learning, we only have single samples without any explicit targets, while in supervised learning we are given input and target output pairs. Due to the lack of target data, the goal of unsupervised learning is to identify in some sense interesting and useful properties of the data itself [32, 34]. This type of learning is usually associated with humans and animals and more accessible than supervised learning since no labeled data is required [34]. As before we model the sampling of the data points by a complete probability space $(\Omega, \mathfrak{F}, \mathbb{P})$. Ω denotes the underlying sampling space, $\mathfrak{F}$ the associated σ-algebra and $\mathbb{P}$ is the probability measure. We consider a random variable $X : \Omega \to \mathcal{X}$ associated with the distribution $\mathcal{T}$, where $\mathcal{X}$ defines the sample space, which is typically a n-dimensional vector space such as $\mathcal{X} \subset \mathbb{R}^n$. Then, each realization of the random variable $X \sim \mathcal{T}$ defines a sample $x \in \mathcal{X}$.

A prototypical example of unsupervised learning is density estimation. In this task, we would like to develop a parametric model that approximates the probability density function of the data such that we can either sample from the corresponding distribution or quantify the probability of unseen data samples. In detail, let x be a sample of the unknown distribution $\mathcal{T}$. We would like to fit a parametric model $p : \mathcal{X} \times \Theta \to [0, \infty)$ that best approximates the probability density function associated with the data. A common approach to estimate the model parameters $\theta \in \Theta$ is to maximize the likelihood of the data given the parameters, which amounts to minimizing the negative log-likelihood

$$\mathrm{J}(\theta) \coloneqq \mathbb{E}_{x \sim \mathcal{T}} \left[-\log p(x, \theta) \right].$$

In contrast to supervised learning, there is no loss function involved that compares the model output to a target output. Figure 3.6 depicts the estimated density of the petal length and width of the iris dataset [126] using a Gaussian mixture model (GMM) [32, Section 9.2] with 2 components. The density of the black data points can be well approximated by the GMM and it is highest at the cluster in the lower left corner.

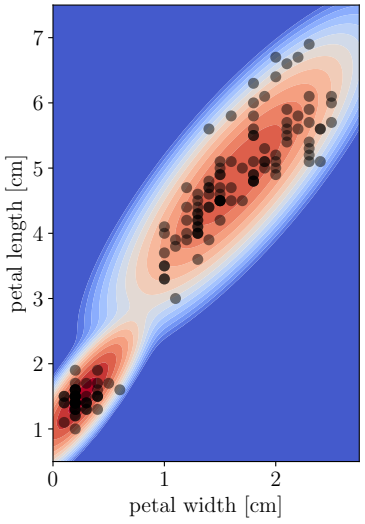


Figure 3.6: Estimated density of the iris dataset [126] in the petal width and petal length plane using two components in a Gaussian mixture model. The estimated probability density function is depicted using a contour plot where warm colors indicate regions of high density.

Another characteristic example of unsupervised learning is clustering. Assume we know that there are K clusters in the data. Then, clustering is the process of partitioning the sample space $\mathcal{X}$ into K disjoint regions called clusters such that the inter-point distance $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ between two samples within a cluster is small. Frequently, the Euclidean distance $d(x, y) = \|x - y\|_2$ is chosen to determine the clusters. The result of the simple K-means clustering algorithm [127, 128] to partition the space spanned by the petal length and width of iris flowers [126] is illustrated in Figure 3.7. Each sample x (dot) is related to one of the $K = 3$ cluster centers μ_j (big X) based on the Euclidean distance to the cluster centers. Thus, the

cluster label $z \in \{1, \dots, K\}$ of a sample point x is determined by

$$z = \underset{j=1,\dots,K}{\operatorname{argmin}} \left\| x - \mu_j \right\|_2 ,$$

given the cluster means μ_j for $j = 1, \dots, K$.

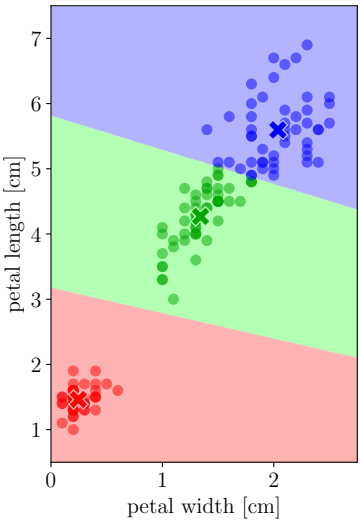


Figure 3.7: Result of applying the K-mean clustering algorithm [127, 128] to the iris dataset [126] for $K = 3$ cluster centers. The cluster centers are indicated by the large X markers, while the samples of the dataset are visualized by dots. The colors indicate the cluster areas as a function of the petal length and width.

3.1.3 Generative and Discriminative Learning

Next, we elaborate on the difference between generative and discriminative learning approaches. Let us consider a continuous, supervised learning problem defined on the underlying complete probability space $(\Omega, \mathfrak{F}, \mathbb{P})$. The realizations of the input random variable $X : \Omega \to \mathcal{X}$ are inputs $x \in \mathcal{X} \subset \mathbb{R}^n$ and the target outputs $y \in \mathcal{Y} \subset \mathbb{R}^m$ are realizations of the target output random variable $Y : \Omega \to \mathcal{Y}$. Both random variables follow the corresponding joint distribution $(X, Y) \sim \mathcal{T}$ and we denote by $\mathcal{T}_\mathcal{X}$ and $\mathcal{T}_\mathcal{Y}$ the marginal distribution of X and Y, respectively.

Due to the definition of conditional probabilities, there are two ways to fit a parametric model to the joint density of the inputs and outputs $p(x, y)$ [113]. The *discriminative model* amounts to

$$p(x, y) \approx p_{\text{dis}}(x, y, \theta_{\text{dis}}) = p_{Y|X}(y|x, \theta_{Y|X}) p_X(x, \theta_X),$$

where $p_{Y|X}(y|x, \theta_{Y|X})$ is a parametric function that approximates the posterior density $p_{Y|X}(y|x)$ and $p_X(x, \theta_X)$ approximates the density of the inputs $p_X(x)$. The parameters of both parts of the discriminative model are aggregated into $\theta_{\text{dis}} = (\theta_{Y|X}, \theta_X)$, which are frequently estimated by maximizing the joint likelihood, resulting in the corresponding loss function

$$J(\theta_{\text{dis}}) = \mathbb{E}_{(X,Y)\sim\mathcal{T}} \left[-\log p_{Y|X}(\cdot\,|\,\cdot, \theta_{Y|X}) \right] + \mathbb{E}_{X\sim\mathcal{T}_\mathcal{X}} \left[-\log p_X(\cdot, \theta_X) \right].$$

Then, the optimization of the parameters decouples. While θ_X can be determined by minimizing the negative log likelihood over the input data (unsupervised learning), the parameters $\theta_{Y|X}$ of the posterior model $p_{Y|X}$ are determined by supervised learning. Consequently, the *discriminative learning* approach separates learning of the input density from learning the posterior density. After learning its parameters, the parametric posterior density $p_{Y|X}$ reflects the likelihood of an output $y \in \mathcal{Y}$ given the input $x \in \mathcal{X}$. The parametric posterior density is used to infer the best output for a given loss [32, 34, 113]. For instance, the zero-one loss, which assigns a constant penalization to every deviation from the true output and zero else, leads to the maximum a-posteriori (MAP) estimate [32, 34] that maximizes the posterior density

$$\widehat{y}(x) \in \underset{y\in\mathcal{Y}}{\operatorname{argmax}}\, p_{Y|X}(y|x, \theta_{Y|X}).$$

As a result, the inference of the associated output for a given input $x \in \mathcal{X}$ is solely determined by the parametric posterior distribution in the *discriminative* approach. The advantage of the discriminative approach is that the estimator is just defined using the parametric posterior, which is often simpler to learn [34, 113]. However, incorporating domain knowledge, which is important in inverse problems, is typically easier in generative approaches.

The *generative model* approximates the joint density by

$$p(x, y) \approx p_{\text{gen}}(x, y, \theta_{\text{gen}}) = p_{X|Y}(x|y, \theta_{X|Y}) p_Y(y, \theta_Y),$$

where $p_{X|Y}(x|y, \theta_{X|Y})$ is a parametric function that estimates the input likelihood density $p_{X|Y}(x|y)$ and $p_Y(y, \theta_Y)$ approximates the density of the outputs $p_Y(y)$. The parameters of both parts of the generative model are subsumed into $\theta_{\text{gen}} =$

$(\theta_{X|Y}, \theta_Y)$. Then, the maximum likelihood estimation of the parameters leads to the loss function

$$J(\theta) = \mathbb{E}_{(X,Y)\sim\mathcal{T}}\left[-\log p_{X|Y}(\cdot\,|\,\cdot\,, \theta_{X|Y})\right] + \mathbb{E}_{Y\sim\mathcal{T}_{\mathcal{Y}}}\left[-\log p(\cdot\,, \theta_Y)\right],$$

which also decouples into a supervised learning problem of the input likelihood $p_{X|Y}$ and an unsupervised learning problem over the output data of the prior density p_Y. Hence, *generative learning* separates the parameter estimation of the likelihood from the prior. Once the parameters $\theta_{X|Y}$ and θ_Y are learned, we can deduce the posterior density using Bayes' Theorem 2.2.7 such that

$$p_{Y|X}(y|x, \theta_{gen}) := \frac{p_{X|Y}(x|y, \theta_{X|Y})p_Y(y, \theta_Y)}{\int_{\mathcal{Y}} p_{X|Y}(x|y', \theta_{X|Y})p_Y(y', \theta_Y)\mathrm{L}^m(\mathrm{d}y')}. \tag{3.5}$$

The corresponding MAP estimate for a given input $x \in \mathcal{X}$ is then given by

$$\widehat{y}(x, \theta_{\text{gen}}) \in \operatorname*{argmax}_{y\in\mathcal{Y}} p_{Y|X}(y|x, \theta_{gen}) = \operatorname*{argmax}_{y\in\mathcal{Y}} p_{X|Y}(x|y, \theta_{X|Y})p_Y(y, \theta_Y).$$

Consequently, the *generative* approach amounts to determining the input likelihood and prior separately and using Bayes' theorem to derive the posterior. The advantage of the generative model is that the input likelihood $p_{X|Y}$ can be easily used to incorporate domain knowledge about the problem, but the inference is based on the associated posterior, whose parameters θ_{gen} have not been trained to discriminate suitable outputs [34, 113].

Not all supervised learning methods can be strictly separated into generative and discriminative approaches, there are also hybrid approaches [113]. For example, we could use a parametric generative model and Bayes' Theorem 2.2.7 to define the associated posterior density, see (3.5). Instead of determining the parameters of the input likelihood and prior separately as in generative learning, we could determine the parameters of the generative model such that the MAP estimator best explains the target outputs $y \in \mathcal{Y}$ given some loss $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. This results in the bilevel optimization problem

$$\min_{\theta_{\text{gen}}\in\Theta} \mathbb{E}_{(X,Y)\sim\mathcal{T}}\left[\ell(\widehat{y}(x, \theta_{\text{gen}}), y)\right] \quad \text{s.t.} \quad \widehat{y}(x, \theta_{\text{gen}}) \in \operatorname*{argmax}_{y'\in\mathcal{Y}} p_{Y|X}(y'|x, \theta_{\text{gen}}),$$

where Θ is the space of feasible parameters. Hence, the parameters θ_{gen} of the *generative* model are determined by *discriminative* learning of the corresponding posterior density. A particular advantage of this approach is that the generative model allows to incorporate domain knowledge into the input likelihood $p_{X|Y}$, which is of particular interest for inverse problems, and the discriminative learning of the parameters directly targets the inference [113].

3.2 Data, Generalization and Model Complexity

In this section, we define the term generalization in machine learning and illustrate its relation to the model complexity and the finite number of data samples.

3.2.1 Finite Data and Generalization

In many applications, we do not have the possibility to constantly draw samples from the data distribution $\mathcal{T}$ due to time or budget constraints. For instance, let us consider the task of cell segmentation of hematoxylin and eosin stained tissue sections. The manual annotation of each sample requires intensive user interactions of an expert pathologist, which comes along with high temporal and monetary costs. Thus, we

cannot use the expected loss (3.1) to identify the model parameters. Nevertheless, we are often able to collect a dataset $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\} \subset \mathcal{X} \times \mathcal{Y}$ consisting of $N \in \mathbb{N}$ sample pairs in this supervised case. Using this dataset we are able to approximate the expected loss (3.1) by the *empirical risk*

$$\mathrm{J}_{\mathrm{emp}}(\theta, \mathcal{D}) \coloneqq \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(f(x, \theta), y), \tag{3.6}$$

which is the average loss across the entire dataset. Then, the best parameters are obtained by minimizing the empirical risk (3.6) for a finite dataset $\mathcal{D}$, which can be done for suitable models and loss functions, as we have seen in Section 2.4. From this point of view, it seems like machine learning is just a special case of optimization. However, in machine learning, we are not only interested in finding the optimal parameters θ^* of a model for a dataset $\mathcal{D}$. We long for models $f : \mathcal{X} \times \Theta \to \mathcal{Y}$ and their parameters $\theta \in \Theta$ that also work well on *new unseen* data samples $x \in \mathcal{X}$, $(x, \cdot) \notin \mathcal{D}$. This desired ability is called *generalization* [115].

In many real-world applications such as autonomous driving or cancer detection, machine learning models have to be robust against all sorts of data variations in order to *generalize* to unseen data samples. Following [129], we call the absolute difference between the expected loss and the empirical risk the *generalization error*, i.e.

$$\mathrm{G}(\theta, \mathcal{D}) = |\mathrm{J}(\theta) - \mathrm{J}_{\mathrm{emp}}(\theta, \mathcal{D})|.$$

If the generalization error is small, we can expect that our model works well on unseen data. Since we cannot compute the expected loss $\mathrm{J}(\theta)$ in many cases, we have to estimate the generalization ability of a model. This is frequently done by partitioning the dataset $\mathcal{D}$ into disjoint train and test datasets [34, 115], such that

$$\mathcal{D} = \mathcal{D}_{\mathrm{train}} \cup \mathcal{D}_{\mathrm{test}} \quad \text{and} \quad \mathcal{D}_{\mathrm{train}} \cap \mathcal{D}_{\mathrm{test}} = \emptyset.$$

Then, the model parameters are estimated by minimizing the empirical risk over the training dataset, i.e.

$$\theta^* = \operatorname*{argmin}_{\theta\in\Theta} \mathrm{J}_{\mathrm{emp}}(\theta, \mathcal{D}_{\mathrm{train}}).$$

Finally, we can compare some error measure (e.g. mean squared error, peak signal-to-noise ratio, ...) on the train and test dataset to see if the model generalizes to the unseen test data. We will see in the next section that the number of samples in the train and test set $N_{\mathrm{train}} = |\mathcal{D}_{\mathrm{train}}|$ and $N_{\mathrm{test}} = |\mathcal{D}_{\mathrm{test}}|$ must be rather large in order to represent the distribution $\mathcal{T}$ sufficiently well.

3.2.2 Model Complexity and Generalization

The *complexity* of a model defines the range of functions that can be well approximated by suitable choices of its parameters [32]. Intuitively speaking, the more complex a model, the richer are its representable functions. Models with low complexity may not be able to explain the functional dependency between input and output samples, while models with large complexity may focus on particular properties of samples. The first effect is known as *underfitting* and the second as *overfitting* [115]. Clearly, we have to avoid both effects in order to develop suitable models for a particular task.

In the following example, we illustrate the underfitting and overfitting effect for a model of different complexity and show that these effects are tightly coupled with generalization. We consider the supervised regression task of fitting a polynomial of degree $d \in \mathbb{N}$ as defined in (3.2). We estimate the parameters of the polynomial $\theta \in \mathbb{R}^{d+1}$ by minimizing the empirical risk (3.6) over the train

dataset $\mathcal{D}_{\text{train}} = \{(x_1, y_1), \ldots, (x_{N_{\text{train}}}, y_{N_{\text{train}}})\}$ using the quadratic loss function

$$\ell(y, \widehat{y}) = (y - \widehat{y})^2.$$

The resulting quadratic training problem can be solved in closed form and its solution is given by

$$\theta = (X_{\text{train}}^\top X_{\text{train}})^{-1} X_{\text{train}}^\top y_{\text{train}} \tag{3.7}$$

in analogy to (3.4). Here, $y_{\text{train}} \in \mathbb{R}^{N_{\text{train}}}$ denotes the vector whose elements are the target outputs of all training samples and $X_{\text{train}} \in \mathbb{R}^{N_{\text{train}} \times d+1}$ is the feature matrix of the training sample inputs as defined in (3.3).

First, we assume that the train dataset is limited to $N_{\text{train}} = 15$ training samples, which are illustrated as black dots in the top plot of Figure 3.8. For this rather small train dataset, we fit polynomials of degree $d \in \{0, 1, \ldots, 14\}$ using (3.7). The degree reflects the model complexity since the number of learnable parameters $\theta \in \mathbb{R}^{d+1}$ increases along with the degree of a polynomial. If we choose a small degree such as $d = 2$, our model can only approximate quadratic functions well. For the task at hand, the quadratic polynomial (dashed blue line in the top plot of Figure 3.8) describes the overall shape of the data samples but it is not able to reflect the peak at 0. Moreover, for $d = 2$ we observe a high empirical risk for the train *and* test dataset, see central plot of Figure 3.8. This is a typical case of underfitting because the class of quadratic functions is not rich enough to approximate the target function. Increasing the degree to $d = 4$ leads to a minimum of the empirical risk of both the train and test dataset, see Figure 3.8 central plot. The minimal empirical risk on both datasets indicates that this is the correct model complexity and the corresponding polynomial is depicted as the green curve in the top plot. It describes all 15 training points well without fluctuations between sample points. If we continue to increase the model complexity, we are able to better fit the training samples, reflected by the decreasing empirical risk of the train dataset (dashed blue curve in the central plot). However, the empirical risk of the test dataset (dash-dotted red curve) strongly increases. For instance, the polynomial of degree $d = 14$ (dash-dotted red curve in the top plot) perfectly fits the training data but oscillates strongly between training samples especially at the left and right boundary, which leads to a high error on the test dataset. This is a prototypical example of overfitting. The polynomial of degree $d = 14$ perfectly explains all training samples but does not generalize to unseen test samples.

Second, we assume that the degree of the polynomial model is fixed to $d = 14$ but we are free to draw more samples to increase the train dataset. The bottom plot of Figure 3.8 shows the empirical risk of the train and test dataset as a function of the number of training samples N_{train}. Clearly, the empirical risks of the train and test dataset do not agree for small N_{train}, but for $N_{\text{train}} > 10^3$ both errors seem to converge towards 10^{-3}. As a result, we are able to avoid overfitting for complex models by increasing the train dataset size. Note that we have to increase the train dataset size N_{train} by two to three orders of magnitude for this simple example in order to learn parameters of the polynomial of degree $d = 14$ that work equally well as the ones identified for degree $d = 4$ using only $N_{\text{train}} = 15$ training samples. Hence, it is reasonable to adapt the model complexity for the particular task.

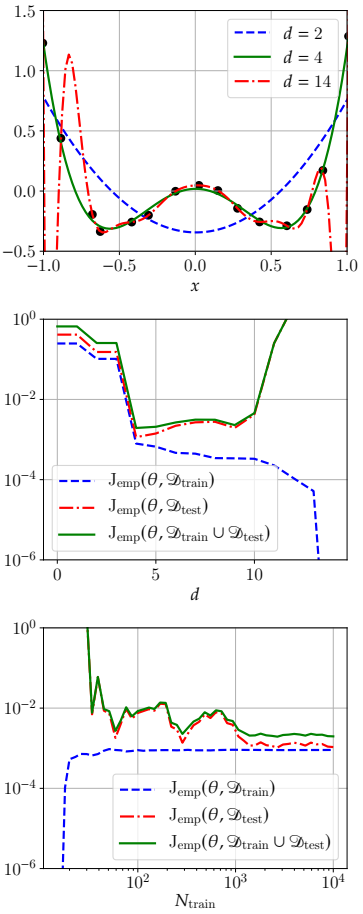


Figure 3.8: The top plot depicts all $N_{\text{train}} = 15$ training samples $(x, y) \in \mathcal{D}_{\text{train}}$ (black dots) along with fitted polynomial of degree $d \in \{2, 4, 14\}$. The middle plot depicts the empirical risk of the train dataset $\mathcal{D}_{\text{train}}$, test dataset $\mathcal{D}_{\text{test}}$, and the entire dataset $\mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{test}}$ as a function of the polynomial degree d using $N_{\text{train}} = 15$. The bottom plot illustrates the empirical risk as a function of the number of training samples N_{train} for fitting a polynomial of degree $d = 14$.

3.3 Neural Networks

In general, neural networks (NNs) are parametric functions that process an input by sequentially applying 'simple' parametric and non-parametric functions [115]. In machine learning, we use these parametric NNs to approximate functions. For example, a NN can be used to approximate a classifier assigning a class label to every

input, or the density of a probabilistic process may be estimated by a NN. Thus, NNs are widely used in many fields such as engineering, finance, and medicine. In this section, we explain the concept of NNs, their building blocks, and motivate state-of-the-art deep learning architecture designs.

We denote by $\mathcal{X}$ the input space of the NN and by $\mathcal{Y}$ its corresponding output space. Likewise, Θ defines the space of admissible parameters of a NN. Then, a NN is formally defined as a parametric mapping from the input and parameter space to the output space [32, 115], i.e.

$$f : \mathcal{X} \times \Theta \to \mathcal{Y}.$$

The NN f is typically highly structured and consists of a composition of simple functions $f_i : \mathcal{X}_{i-1} \to \mathcal{X}_i$ for $i = 1, \ldots, l$

$$f = f_l \circ f_{l-1} \circ \cdots \circ f_i \circ \cdots \circ f_2 \circ f_1,$$

which we call *building blocks* or *layers* of NNs. The combination of the building blocks defines the *architecture* of a NN and the length of the entire sequence of layers $l \in \mathbb{N}$ is called the *depth*. Each layer f_i is a (non)linear and (non)-parametric function that maps its input $x_{i-1} \in \mathcal{X}_{i-1} \subset \mathbb{R}^{n_{i-1}}$ to the output $x_i \in \mathcal{X}_i \subset \mathbb{R}^{n_i}$ via

$$x_i = f_i(x_{i-1}),$$

where the initial space $\mathcal{X}_0$ is defined by the input space $\mathcal{X}_0 = \mathcal{X}$. The last layer $f_l : \mathcal{X}_{l-1} \to \mathcal{X} = \mathcal{Y}$ of the sequence is called the *output* layer, while the remaining layers are called *hidden* layers. Each layer f_i typically maps into a vector space $\mathcal{X}_i \subset \mathbb{R}^{n_i}$ and the dimension n_i defines the *width* of a layer [115]. The resulting function is called a *neural* network because it loosely resembles the synaptic connections within the human brain [130].

So far, a NN f simply defines a mapping from inputs x and parameters θ to outputs y. In order to adapt a NN to a particular task, we need to *learn* its parameters θ. In supervised learning, for instance, we could determine them by minimizing a cost function such as (3.1) using a suitable first-order method (Section 2.4.3) for given task-specific samples from the input and output space.

3.3.1 Building Blocks

In this section, we introduce essential building blocks of NNs, where we distinguish between linear and nonlinear functions.

3.3.1.1 Linear Layers

Let $x_{i-1} \in \mathcal{X}_{i-1} \subset \mathbb{R}^{n_{i-1}}$ be an n_{i-1}-dimensional real vector, which is the input to the i^{th} layer. The output of a linear layer is computed by applying the affine transform $\mathrm{W}^l : \mathcal{X}_l \to \mathcal{X}_{i+1}$

$$x_i = \mathrm{W}^i(x^i) = W^i x_{i-1} + b^i,$$

where $\mathcal{X}_i \subset \mathbb{R}^{n_i}$ is the output space and the number $n_i \in \mathbb{N}$ defines the *width* of the layer. The parameters of the layer θ_i are encoded in the *weight* $W^i \in \mathbb{R}^{n_i \times n_{i-1}}$ and the *bias* $b^i \in \mathbb{R}^{n_i}$.

A linear layer is called a *fully connected* layer if all elements of the weight matrix $W^i = (w^i_{j,k})$ and the bias vector $b^i = (b^l_j)$ for $j = 1, \ldots, n_i$ and $k = 1, \ldots, n_{i-1}$ are learnable. Then, both the weight and the bias are entirely included in the layer's parameters, i.e. $\theta_i = (W^i, b^i)$. In this case, n_i is also frequently called the *number of neurons* of a layer.

A very important class of linear layers that are frequently used in signal processing are *convolution* layers. In contrast to the fully connected layer, not all elements of the weight matrix W^i and the bias vector b^i are learnable. Let us consider an input image of size $n_1 \times n_2$ with c_{i-1} feature channels, i.e. $x_{i-1} \in \mathbb{R}^{nc_{i-1}}$ with $n = n_1 \cdot n_2$. The output of a convolutional layer $x_i \in \mathbb{R}^{nc_i}$ is a signal with the same length and c_i feature channels. Then, the output is determined by applying $c_i \cdot c_{i-1}$ convolution filters $k^i_{u,v} \in \mathbb{R}^{s \times s}$ with spatial support $s \times s$ to the input plus a bias $b^i_u \in \mathbb{R}$ for $u = 1, \ldots, c_i$ and $v = 1, \ldots, c_{i-1}$. Let $X^{i-1,v} \in \mathbb{R}^{n_1 \times n_2}$ represent the v^{th} channel of the input image. The two-dimensional discrete convolution of this input channel with a convolution kernel $k^i_{u,v}$ is defined as

$$(X^{i-1,v} * k^i_{u,v})_{l.m} = \sum_{o=1}^{s} \sum_{p=1}^{s} X^{i-1,v}_{l-o+\lfloor s/2 \rfloor, m-p+\lfloor s/2 \rfloor} \cdot (k^i_{u,v})_{(o,p)},$$

where an appropriate boundary handling such as symmetric padding is applied if $0 < l - o + \lfloor s/2 \rfloor < n_1$ and $0 < m - p + \lfloor s/2 \rfloor < n_2$ do not hold. This convolution operation can be represented by a matrix-vector product

$$X^{i-1,v} * k^i_{u,v} \iff K^i_{u,v} x^v_{i-1},$$

where $K^i_{u,v} \in \mathbb{R}^{n \times n}$ is a Toeplitz matrix [131] whose diagonal elements are determined by the corresponding filter $k^i_{u,v}$, and $x^v_{i-1} \in \mathbb{R}^n$ is the vector representation of the v^{th} input channel, see Section 4.1.1. Then, the resulting weight matrix and bias vector of a convolution layer are given by

$$W^i = \begin{pmatrix} K^i_{1,1} & K^i_{1,2} & \cdots & K^i_{1,c_l} \\ K^i_{2,1} & K^i_{2,2} & \cdots & K^i_{2,c_l} \\ \vdots & \vdots & \ddots & \vdots \\ K^i_{c_{l+1},1} & K^i_{c_{l+1},2} & \cdots & K^i_{c_{l+1},c_l} \end{pmatrix} \quad \text{and} \quad b^i = \begin{pmatrix} \mathbb{1}_n b^i_1 \\ \mathbb{1}_n b^i_2 \\ \vdots \\ \mathbb{1}_n b^i_{c_{l+1}} \end{pmatrix},$$

respectively, where $\mathbb{1}_n \in \mathbb{R}^n$ is a vector whose entries are equal to 1. The parameters of a convolution layer are all convolution filters and bias weights $\theta^i = (k^i_{u,v}, b^i_u,$ for $u = 1, \ldots, c_i,$ and $v = 1, \ldots, c_{i-1})$.

Further important building blocks that are frequently linked with convolution layers are *down-* and *upsampling* layers. These layers are used to decrease and increase the spatial support of a signal. Hence, downsampling layers can be used to compress or encode local information and upsampling layers are used for decoding. We again consider an input image $x_{i-1} \in \mathbb{R}^{n_{i-1}c_{i-1}}$ of size $n_{i-1} = n_1^{i-1} \times n_2^{i-1}$ with c_{i-1} channels. The output of an s-fold downsampling layer has the same number of channels as the input but the spatial size is reduced by copying only every s^{th} pixel in the horizontal and vertical direction to the output $x \in \mathbb{R}^{n_i c_{i-1}}$, as illustrated by the red arrows in Figure 3.9. Thus, $n_i = n_{i-1}/s^2$. In contrast, an s-fold upsampling layer computes its output by copying every s^{th} pixel from the input image, while the remaining pixels are set to 0, which is visualized by the blue arrows in Figure 3.9. Then, its output is an image of size $n_i = n_{i-1}s^2$. As a result, all entries of the associated weight matrices are either 0 or 1, i.e. $W^i \in \{0,1\}^{n_i c_{i-1} \times n_{i-1} c_{i-1}}$, and the bias vector vanishes: $b^l = 0 \in \mathbb{R}^{n_i c_{i-1}}$. Hence, there are no learnable parameters in a down- or upsampling layer. Note that the combination of a convolution and downsampling layer is called a *strided convolution* layer while the combination of a convolution and upsampling layer is called a *transposed convolution* layer [79, 80].

There are many more linear layer types such as *dilated convolution* layers [132] and *padding* layers, which are established in the machine learning community. We refer the interested reader to the documentation of Tensorflow [79] and PyTorch [80] for a recent list.

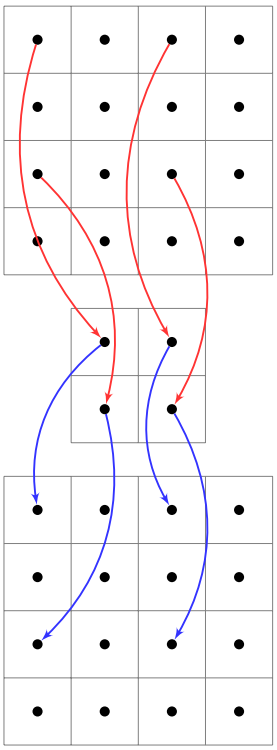


Figure 3.9: Visualization of the down- and upsampling operation. We show the pixel grids of three images. The black dots visualize the pixel centers and the gray lines the pixel outline. The red arrows illustrate a 2-fold downsampling of the upper image to the central image while the blue arrows symbolize the copying of the pixels from the central image to the 2-fold upsampled bottom image.

3.3.1.2 Nonlinear Layers

Nonlinear layers are the essential ingredients that bring NNs up and running. In fact, a NN without any nonlinear layer can only represent linear functions because the resulting function is linear. However, many realistic problems discussed in the previous sections can only be tackled by learning a nonlinear parametric function.

In contrast to linear layers, nonlinear layers are hardly used to reduce or increase the width within a NN. Let $x^l \in \mathcal{X}_l \subset \mathbb{R}^{n_l}$ represent the input of a layer. The output $x^{l+1} \in \mathcal{X}_{l+1} = \mathcal{X}_l$ of a nonlinear *activation layer* is computed by applying the function

$$\Phi : \mathcal{X}_l \to \mathcal{X}_l,\ (x_1^l, \dots, x_n^l) \mapsto (\phi(x_1^l), \dots, \phi(x_n^l))$$

to the input x^l, which amounts to a transformation of each input element by the nonlinear *activation function* $\phi : \mathbb{R} \to \mathbb{R}$. Figure 3.10 visualizes common activation functions used in NNs. The hyperbolic tangent tanh and the logistic sigmoid

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

are frequently used in NNs since both functions are smooth and monotonically increasing. However, the tanh and sigmoid activation functions saturate for $|x| > 3$. This saturation can make gradient-based learning of the network parameters hard because the gradient of the tanh and sigmoid function vanishes within the saturated regions. The rectified linear unit [133, 134] defined as

$$\mathrm{ReLU}(x) := \max(x, 0)$$

does not saturate for positive inputs and leads to sparse features [135] since all negative values are mapped to 0. The caveat of the ReLU function is its non-differentiability at 0, which is often ignored in practice and the subgradient $0 \in \partial\mathrm{ReLU}(0)$ is used. Nevertheless, today it is the default nonlinear activation function applied in deep neural networks [115].

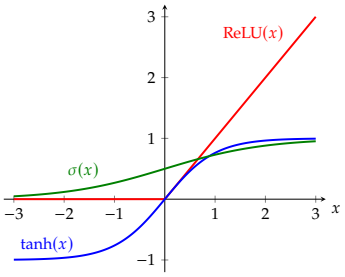


Figure 3.10: Plot of typical nonlinear activation functions used in neural networks.

Along with these nonparametric activation functions there also exist parametric activation functions. He et al. [82] proposed the parametric ReLU (PReLU) function

$$\mathrm{PReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ ax & \text{if} x \leq 0 \end{cases}$$

where the slope $a \in \mathbb{R}$ of the negative orthant is learnable. In addition, radial basis functions

$$\phi_{\mathrm{RBF}}(x) = \sum_{i=1}^{N_w} w_i \varphi(x - \mu_i)$$

with $\varphi(x) = \exp\left(\frac{-x^2}{2\sigma^2}\right)$ have been used as activation functions in NNs [136]. Here, the weights of the basis functions $w_i \in \mathbb{R}$, the centers $\mu_i \in \mathbb{R}$ and the factor $\sigma \in \mathbb{R}$ can be possibly learned and N_w defines the number of radial basis functions.

So far, every nonlinear layer applied the same scalar activation function to all elements of its input. *Normalization* layers are a class of nonlinear layers that combine all or a fraction of the input elements to compute the output. Let $x \in \mathcal{X} \subset \mathbb{R}^n$ be the input of a layer. The *softmax* layer maps an input $x \in \mathcal{X}$ to an output $y \in \Delta^n$ on the unit simplex Δ^n and is defined as

$$y_i = \mathrm{softmax}(x)_i = \frac{\exp(x_i)}{\sum_{j=1}^n \exp(x_j)}$$

for $i = 1, \dots, n$. It can be used to transform an input feature x into a discrete probability distribution with n possible outcomes [115]. Thus, it is frequently applied as the

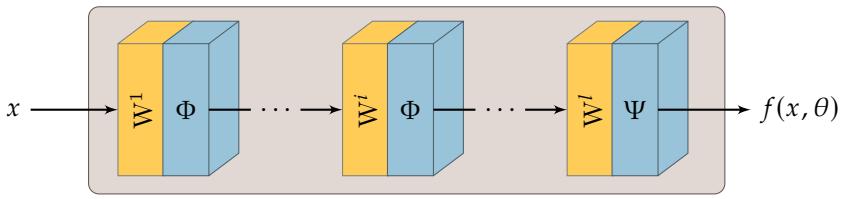


Figure 3.11: Illustration of a feed-forward neural network. The input x is processed by a sequence of layers consisting of linear W_i and nonlinear activation layers Φ. The final output is generated by applying the nonlinear activation layer Ψ.

output layer of a NN designed for multi-class classification problems. Another variant of normalization layers is called *batch normalization* [137], which was introduced to reduce the internal variance in deep NNs such that their gradient-based training becomes simpler. Later also *instance normalization* [138] and *group normalization* [139] layers were introduced. All three layer types estimate the mean and variance of a subset of the elements of the layer's input in order to produce a normalized output. For more details on these normalization layers, we refer the interested reader to the corresponding papers [137–139].

3.3.2 Deep Learning Architectures

The combination of the building blocks introduced in the previous section defines the architecture of a NN. In this section, we discuss basic architectures of NNs and motivate state-of-the-art NN design patterns.

3.3.2.1 Feed-forward Networks

The feed0forward NNs architecture is the prototypical design of NNs. Such networks are also known as multilayer perceptrons [32, 115] and basically consist of an alternating chain of fully connected layers and nonlinear activation layers, i.e.

$$f = \Psi \circ \mathrm{W}_l \cdots \Phi \circ \mathrm{W}_2 \circ \Phi \circ \mathrm{W}_1,$$

which is illustrated in Figure 3.11. Typically, all nonlinear hidden layers in a NN use the same activation layer Φ. While the dimension of the input layer is defined by the input space $\mathcal{X} \subset \mathbb{R}^n$, the choice of the width of the hidden layers is free and often heuristically determined.

The dimension of the output layer and the output activation layer Ψ depend on the specific problem. For the task of approximating a scalar-valued function, for instance, the output space is $\mathcal{Y} = \mathbb{R}$ and a suitable output activation function is the identity mapping $\Psi(x) = x$, hence $\mathcal{X}^l = \mathbb{R}$. In the case of binary classification, we frequently learn a NN to predict the posterior probability of a class given the input. Thus, $\mathcal{Y} = [0, 1]$ and the output activation layer applies the sigmoid function, i.e. $\Psi(x) : \mathbb{R} \to [0, 1]$, $x \mapsto \sigma(x)$. Finally, we can design a NN for a multi-label classification problem. Here, the NN predicts for each input a discrete probability distribution over the class label $\{1, \ldots, C\}$. Then, the output space is $\mathcal{Y} = \Delta^C \subset \mathbb{R}^C$ and we can use the softmax function to map the output features $x^l \in \mathcal{X}_l \subset \mathbb{R}^C$ to the C-dimensional unit simplex.

Next, we present a result from approximation theory of NNs. We consider a feed-forward NN $f : \mathcal{X} \subset \mathbb{R}^n \times \Theta \to \mathbb{R}$ with a single nonlinear hidden layer, i.e.

$$f(x, \theta) = \mathrm{W}^2 \Psi(\mathrm{W}^1 x), \tag{3.8}$$

where $x \in \mathbb{R}^n$, $\mathrm{W}^1 : \mathbb{R}^n \to \mathbb{R}^{n_1}$ and $\mathrm{W}^2 : \mathbb{R}^{n_1} \to \mathbb{R}$ are linear layers, and Φ is a nonlinear activation layer applying a nonlinear function $\phi : \mathbb{R} \to \mathbb{R}$ to every input element. Then, the NN is parameterized by $\theta = (W^1, b^1, W^2, b^2) \in \Theta$.

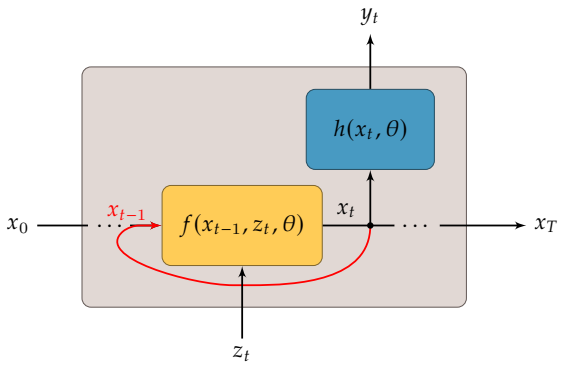


Figure 3.12: Illustration of the basis principle of a recurrent neural network. A sequence of states x_t is generated from the current input z_t and the previous state x_{t-1}. The output at time y_t is a nonlinear transformation of the current state x_t. The red arrow visualizes the feedback of the output state towards the input state.

Theorem 3.3.1 (Universal Approximation) *Let $\mathcal{X} \subset \mathbb{R}^n$ be a compact set. Suppose $f : \mathcal{X} \times \Theta \to \mathbb{R}$ is a feed-forward NN as defined in (3.8) and let $\phi : \mathbb{R} \to \mathbb{R}$ be a nonconstant, bounded and monotonically increasing and continuous function. Then for an arbitrary $\epsilon > 0$, there exist a width $n_1 \in \mathbb{N}$ and parameters $\theta \in \Theta$ such that*

$$\max_{x \in \mathcal{X}} |f(x, \theta) - \widehat{f}(x)| < \epsilon$$

for any $\widehat{f} \in C^0(\mathcal{X}, \mathbb{R})$.

Proof. See [140, Theorem 1]. □

Thus, a feed-forward NN can approximate any continuous function over a compact set with arbitrary precision given a large enough width n_1.

If all parametric linear layers within a NN are convolution layers, we call it a convolutional neural network (CNN). One can think of a CNN as applying a corresponding NN to the local neighborhood at every position of a signal. Thus, CNNs can be used to process an entire signal of arbitrary length — only limited by the amount of available memory. An input signal $\mathbb{R}^{nc}$ of length n with c channels is processed by alternatingly applying convolution and nonlinear activation layers. In contrast to NNs, the width of a layer in a CNN is determined by the number of its output feature channels. CNNs are translation invariant and widely used in imaging to, for instance, semantically segment [141], denoise [142] or classify [143] images.

3.3.2.2 Recurrent Networks

In NNs and CNNs the information 'flows' from the input through each layer towards the output and each layer transforms its input information only once. Thus, there is no *feedback* from the output of a layer towards its input. Neural networks that include feedback connections are called recurrent neural networks (RNNs) [115, 144]. While CNNs can be used to extract meaningful information from entire signals, RNNs are designed to process signals sequentially. Let $z_t \in \mathcal{Z} \subset \mathbb{R}^l$ for $t = 1, \dots, T$ be an input signal of length T. An RNN can be interpreted as a dynamical system with a state $x_t \in \mathcal{X} \subset \mathbb{R}^n$ of the form

$$\begin{aligned} x_t &= f(x_{t-1}, z_t, \theta), \\ y_t &= h(x_t, \theta), \end{aligned}$$

where the *state transition function* $f : \mathcal{X} \times \mathcal{Z} \times \Theta \to \mathcal{X}$ computes the current state $x_t \in \mathcal{X}$ based on the previous state $x_{t-1} \in \mathcal{X}$ and the current input z_t. At every time step t the output function $h : \mathcal{X} \times \Theta \to \mathcal{Y}$ generates a single element of the output sequence $y_t \in \mathcal{Y} \subset \mathbb{R}^m$, see Figure 3.12. RNNs of this form can compute any function computable by a Turing machine [145]. There are various different types

of RNNs such as LSTMs [146] and neural Turing machines [147]. Since RNNs are specifically designed to process sequential data, they are successfully applied to various realistic tasks that require the extraction of contextual information from a data stream such as language translation [148] or speech recognition [149].

3.3.2.3 Residual Networks

A network architecture that in some sense combines feed-forward and recurrent NN architectures are residual neural networks (ResNets) due to He et al. [150]. As in feed-forward NNs the input information is fed into the network and transformed by each consecutive layer once to extract the output information. Hence, there is no feedback in ResNets but they do advocate *skip connections* to introduce shortcuts in *residual blocks*, see Figure 3.13. He et al. [150] included these skip connections to avoid a vanishing gradient problem [115] in deep feed-forward NNs and thereby ease gradient-based training of the network parameters. The resulting ResNets could be successfully trained with hundreds of layers, which led to a significant performance increase for numerous problems [150].

Once more let us consider an input $x \in \mathcal{X} \subset \mathbb{R}^n$, which is transformed into initial features $x_0 \in \mathcal{F} \subset \mathbb{R}^{n_f}$ by

$$x_0 = \Phi(\mathrm{W}^{\mathrm{in}} x)$$

using a linear input layer $\mathrm{W}^{\mathrm{in}} : \mathcal{X} \to \mathcal{F}$ and a nonlinear activation layer $\Phi : \mathcal{F} \to \mathcal{F}$. These initial features x_0 are then processed by a sequence of *residual blocks* $h_t : \mathcal{F} \to \mathcal{F}$ for $t = 1, \dots, T$, as illustrated in Figure 3.13. There are many different designs for residual blocks [151]. A common one is of the form

$$x_t = h_t(x_{t-1}) = \overline{\Phi}(x_{t-1} + f_t(x_{t-1})),$$

where $\overline{\Phi} : \mathcal{F} \to \mathcal{F}$ can be a nonlinear activation layer or the identity mapping and $f_t : \mathcal{F} \to \mathcal{F}$ is the *residual function*. The residual function is frequently a small feed-forward NN. For instance, in Figure 3.13 the residual function is a two-layer NN with a linear output activation layer, i.e. $f_i = \mathrm{W}^{i,2} \circ \Phi \circ \mathrm{W}^{i,1}$. The output $y \in \mathcal{Y} \subset \mathbb{R}^m$ of a ResNet is commonly computed by applying a linear layer $\mathrm{W}^{\mathrm{out}} : \mathcal{F} \to \mathbb{R}^m$ in conjunction with the output activation layer $\Psi : \mathbb{R}^m \to \mathcal{Y}$ to the final state $x_T \in \mathcal{F}$.

If the last activation layer of each residual block in a ResNet is the identity mapping ($\overline{\Phi} = \mathrm{Id}$), the features $x_t \in \mathcal{F}$ evolve according to the discrete dynamical system

$$x_t = x_{t-1} + f_t(x_{t-1}).$$

E [152] as well as Haber and Ruthotto [153] suggested to interpret this ResNet as a forward Euler discretization of the corresponding non-autonomous ordinary differential equation (ODE)

$$\frac{x_t - x_{t-1}}{\Delta t} \approx \dot{x} = f(t, x).$$

This connection of ResNets and ODEs leads to various network architectures inspired by stable ODE discretization schemes [153–155] and enables a theoretical analysis of ResNets from the ODE and optimal control perspective [58, 156].

We conclude this section by introducing dense neural networks (DenseNets) [157], which are a variant of ResNets. DenseNets extend the idea of skip connections and include all previously computed features $x_i \in \mathcal{F}$ for $i = 0, \dots, t-1$ as input to the t^{th} *dense block* $h_t : \mathcal{F}^t \to \mathcal{F}$, which results in the dynamical system

$$x_t = f_t(x_{t-1}, x_{t-2}, \dots, x_1, x_0).$$

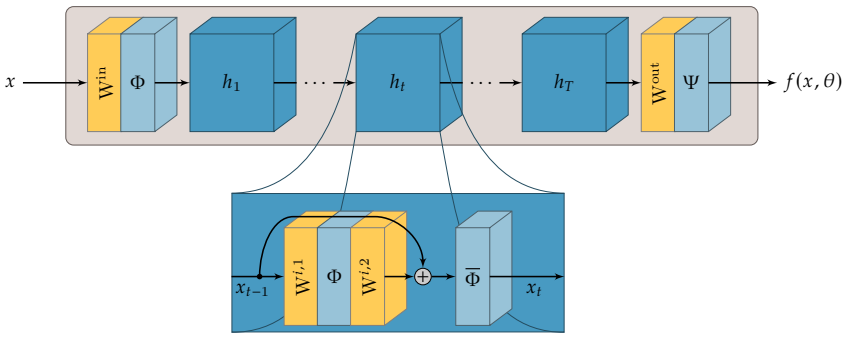


Figure 3.13: Visualization of the residual neural network architecture. The initial features are extracted from the input x using a linear input layer W_{in} and the nonlinear activation layer Φ. These features are then processed by a sequence of residual blocks h_i. The final output is generated by a linear layer W_{out} and the output layer Ψ.

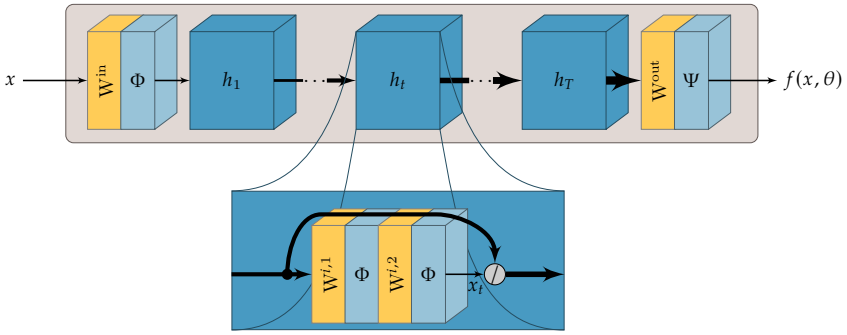


Figure 3.14: Visualization of the dense neural network architecture. Initial features are extracted from the input x using the linear layer W_{in} and the nonlinear activation layer Ψ as in the residual architecture. These features are then processed by a sequence of residual functions h_i that operate on the entire history of features $(x_0, \ldots, x_{i-1})$ and increase the number of features by concatenating (symbolized by /) a nonlinear transformation $(\Psi \circ W_i^2 \circ \Psi \circ W_i^1)$ to the previous features. The increasing number of features is symbolized by the increasing width of the arrows. Finally, the output is generated by a linear layer W_{out} and the output layer Ψ.

The structure of the DenseNet is rather similar to the ResNet architecture, as shown in Figure 3.14. The major difference is that the output of the residual function is not added to the input features but concatenated. This leads to a consistent increase of the feature dimension from the input towards the output of a DenseNet. Therefore, every dense block is able to extract relevant information from the entire history of features, which results in a performance increase on various tasks [157] compared to ResNets. Moreover, the memory footprint and the number of learnable parameters of DenseNets is smaller compared to competing architectures due to the excessive reuse of information encoded in intermediate features [157].

Towards Learning Data-driven Regularizers in Imaging

4 Regularizers in Imaging

As we have seen in Chapter 1 various problems in computer vision and medical imaging can be cast as inverse problems. The task for these problems is to estimate an unknown image y given some observations z that often only partially capture the underlying image and are subject to measurement uncertainties. To account for this loss of information and measurement noise in the observations z, the Bayesian approach to the statistical viewpoint of inverse problems can be used to develop a rigorous framework [31]. In its essence the Bayesian approach advocates to summarize all knowledge about the unknown solution x of an inverse problem using the posterior distribution $p(x|z)$, which reflects the belief in a distinct solution x given the observations z. According to Bayes' theorem [34] the posterior probability $p(x|z)$ is given by the product of the data likelihood $p(z|x)$ and the prior $p(x)$ weighted by the evidence $p(z)$, i.e.

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)}.$$

Here, the likelihood $p(z|x)$ assigns a probability to the observations z given the solution x using an underlying model of the acquisition process, while the prior probability $p(x)$ is determined by a prior model reflecting the general knowledge about suitable solutions. The evidence, which is also known as the marginal likelihood, defines the probability of the observations z and could be computed by marginalizing the joint probability over all possible images $x \in \mathcal{X}$, i.e. $p(z) = \int_{\mathcal{X}} p(z|x)p(x)\,\mathrm{d}x$ [1]. In many applications modeling the likelihood is straight-forward due to the profound physical knowledge about the acquisition process, e.g. low-dose computed tomography (CT), accelerated magnetic resonance imaging (MRI), or image deconvolution. However, the grand challenge in inverse problems for imaging is the development of a prior model that captures the entire complexity of the statistics of natural images. Later in this chapter, we discuss how complex and nonlinear the statistics of natural images actually are.

1: This integral is not numerically tractable in most realistic applications in imaging. For example, consider a small discrete image of size 8×8 and assume each pixel can take values of the discrete set $\{0, \dots, 255\}$. Then, the number of all possible images is $|\mathcal{X}| = 256^{64} = 1.3408 \cdot 10^{154}$. This is much more than the estimate of the total number of protons in our observable universe, which is around 10^{80} due to Arthur S. Eddington [158].

Using the likelihood and prior model, a solution to an inverse problem is frequently computed by maximizing the posterior probability

$$\widehat{x} \coloneqq \operatorname*{argmax}_x p(x|z) = \operatorname*{argmax}_x p(z|x)p(x).$$

The solution $\widehat{x}$ is known as the maximum a-posteriori (MAP) estimator [32, 34]. Note that the evidence $p(z)$ can be omitted since it does not affect the maximizing argument. If we consider the objective function in a negative logarithmic domain, we end up with the following equivalent minimization problem

$$\widehat{x} = \operatorname*{argmin}_x -\log p(z|x) - \log p(x),$$

which can be linked with the variational formulation for inverse problems. The variational formulation amounts to minimizing an energy $\mathrm{E}(x, z)$ composed of a data fidelity term D and a regularizer R, i.e.

$$\widehat{x} = \operatorname*{argmin}_x \left\{ \mathrm{E}(x, z) \coloneqq \mathrm{D}(x, z) + \mathrm{R}(x) \right\}.$$

Thus, the data fidelity term corresponds to the negative log-likelihood $-\log p(z|x)$ and the regularizer can be identified with the negative log-prior $-\log p(x)$. As a result, the data fidelity term models detailed knowledge about the acquisition process

and associates the solution x with the observation z, while the regularizer reflects all the general knowledge about the solution.

Let us consider the inverse problem of CT reconstruction to illustrate the variational approach. Here, the task is to compute an attenuation map $x \in \mathbb{R}^n$ given the observed X-ray projection data $z \in \mathbb{R}^l$, which consists of l ray measurements. Let us assume that the linear projection operator $A \in \mathbb{R}^{l \times n}$ extracts the projection data for a given attenuation map and further assume that the measurement uncertainty is due to independent and identically distributed (i.i.d.) Gaussian noise with variance σ^2, i.e. $z \sim \mathcal{N}(Ay, \sigma^2 \,\mathrm{Id})$, where $y \in \mathbb{R}^n$ is the ground truth attenuation map. Then, the corresponding likelihood is given by

$$p(z|x) = \det\left(2\pi\sigma^2 \,\mathrm{Id}\right)^{-\frac{1}{2}} \exp\left(-\frac{1}{2\sigma^2} \|Ax - z\|_2^2\right).$$

Taking the negative logarithm and neglecting the constant terms results in the corresponding data fidelity term

$$\mathrm{D}(x, z) = \frac{1}{2\sigma^2} \|Ax - z\|_2^2 .$$

Suppose the only piece of information we have about the unknown solution is that each element of the attenuation map is positive[2] and bounded from above by 1 due to some physical limitations. A suitable prior distribution to reflect this circumstance is given by the uniform distribution $x \sim \mathcal{U}(0,1)^n$. Then, the particular form of the prior is defined as

2: In fact a negative attenuation makes no sense from a physical point of view.

$$p(x) = \prod_{i=1}^{n} \mathbb{1}_{[0,1]}(x_i)$$

with

$$\mathbb{1}_{[0,1]}(x) = \begin{cases} 1 & \text{for } 0 \le x \le 1 \\ 0 & \text{else} \end{cases}.$$

Again taking the negative logarithm, we get the corresponding regularizer

$$\mathrm{R}(x) = \sum_{i=1}^{n} \delta_{[0,1]}(x_i)$$

for

$$\delta_{[0,1]}(x) = \begin{cases} 0 & \text{for } 0 \le x \le 1 \\ \infty & \text{else} \end{cases}.$$

By denoting $\delta_{[0,1]^n}$ as the indicator function of the set $[0,1]^n$, the regularizer is given by $R(x) = \delta_{[0,1]^n}(x)$. Thus, computing the MAP estimator for this CT reconstruction problem is equivalent to minimizing the variational energy

$$\widehat{x} = \operatorname*{argmin}_x \frac{1}{2\sigma^2} \|Ax - z\|_2^2 + \delta_{[0,1]^n}(x).$$

This simple problem already demonstrates that the likelihood and the corresponding data fidelity term can be often directly deduced from the problem at hand, whereas modeling the prior and the regularizer requires knowledge about the structure of the solution.

In the remainder of this chapter, we introduce a matrix and vector representation for discrete images and elaborate on the nonlinear and diverse structure of the statistics of natural images. Then, we provide an overview of principles exploited to design classical regularizers and finally present parametric regularizers which can be adapted to specific tasks by learning their parameters from data.

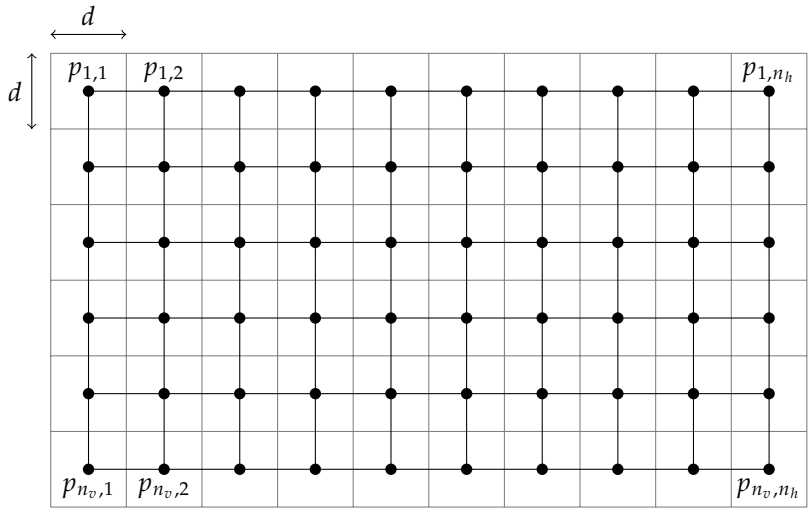


Figure 4.1: Illustration of the pixel grid $P \in \mathbb{R}^{n_v \times n_h}$ of an image of size $n_v \times n_h$. Each pixel has a quadratic shape with edge length d and its center is located at $p_{i,j}$ for $(i,j) \in \mathcal{I}$.

4.1 Representation of Discrete Images

In this section, we define the representations of discrete images used throughout this thesis. First, we define the matrix representation of discrete 2-dimensional images and later link it to the vector representation of images. Finally, the discrete image gradient representing the vertical and horizontal gradients is defined.

4.1.1 Discrete Images

Let $d > 0$ be the distance between two centers of adjacent image pixels and $n_v, n_h \in \mathbb{N}$ the number of vertical and horizontal pixels of an image, respectively. Then, a discrete image of size $n_v \times n_h$ is given by the matrix $X \in \mathbb{R}^{n_v \times n_h}$ and its elements

$$x_{i,j} \in \mathbb{R}, \quad \text{for} \quad (i,j) \in \mathcal{I} \coloneqq \{1, \dots, n_v\} \times \{1, \dots, n_h\}$$

describe the intensity values of the image at the pixel locations

$$p_{i,j} = \begin{pmatrix} id \\ jd \end{pmatrix}$$

within a regular rectangular pixel grid $P \in \mathbb{R}^{n_v \times n_h}$, as illustrated in Figure 4.1. In this formulation, we assumed for the sake of simplicity that the image pixels are quadratic. In the case of rectangular pixels, the distances of the pixels in the horizontal and vertical direction must be selected accordingly.

Beside the matrix representation of an image $X \in \mathbb{R}^{n_v \times n_h}$, we frequently use its vector representation $x \in \mathbb{R}^n$ for $n = n_v \cdot n_h$. The vector representation of a discrete image is simply defined by the lexicographic ordering of the elements of the matrix representation X, i.e.

$$x \coloneqq \begin{pmatrix} x_{1,1} \\ x_{1,2} \\ \vdots \\ x_{1,n_h} \\ x_{2,1} \\ \vdots \\ x_{n_v,(n_h-1)} \\ x_{n_v,n_h} \end{pmatrix}.$$

Further, we often use just a single index $i = 1, \dots, n$ to identify the elements of the vector representation $x = (x_1 \ \dots \ x_n)^\top$ in order to simplify notation. To distinguish

both representations, we use uppercase letters for the matrix representation and lowercase letters for the vector representation.

4.1.2 Discrete Image Gradients

The gradient of discrete images is typically computed by means of finite differences. Thus, we define at every pixel location $p_{i,j}$ with $(i,j) \in \mathcal{I}$ the discrete gradient of a discrete image $X \in \mathbb{R}^{n_v \times n_h}$ by

$$\frac{1}{d}\begin{pmatrix} x_{i,j+1} - x_{i,j} \\ x_{i+1,j} - x_{i,j} \end{pmatrix},$$

where we assume that the gradient vanishes on the boundary (Neumann boundary conditions). In order to define this gradient in the matrix representation, we use the horizontal and vertical gradient matrices $DX_h, DX_v \in \mathbb{R}^{n_v \times n_h}$ with elements computed by

$$(DX_h)_{i,j} = \begin{cases} \frac{1}{d}(x_{i,j+1} - x_{i,j}) & \text{for } 1 \le j < n_h \\ 0 & \text{for } j = n_h \end{cases}$$

and

$$(DX_v)_{i,j} = \begin{cases} \frac{1}{d}(x_{i+1,j} - x_{i,j}) & \text{for } 1 \le i < n_v \\ 0 & \text{for } i = n_v \end{cases},$$

respectively.

In analogy to the vector representation of images, we define the vector representation of vertical and horizontal image gradients by the lexicographic ordering of the associated matrix representations. Thus, the vector representations of the horizontal and vertical image gradient are given by

$$D_h x \coloneqq \frac{1}{d}\begin{pmatrix} x_{1,2} - x_{1,1} \\ x_{1,3} - x_{1,2} \\ \vdots \\ x_{1,n_h} - x_{1,(n_h-1)} \\ 0 \\ x_{n_v,2} - x_{n_v,1} \\ \vdots \\ x_{n_v,n_h} - x_{n_v,(n_h-1)} \\ 0 \end{pmatrix} \quad D_v x \coloneqq \frac{1}{d}\begin{pmatrix} x_{2,1} - x_{1,1} \\ x_{2,2} - x_{1,2} \\ \vdots \\ x_{2,n_h} - x_{1,n_h} \\ x_{3,1} - x_{2,1} \\ \vdots \\ x_{n_v,n_h} - x_{(n_v-1),n_h} \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

These equations define the linear matrix operators $D_h, D_v \in \mathbb{R}^{n \times n}$ that extract the horizontal respectively vertical gradient of an image represented by the vector $x \in \mathbb{R}^n$. In addition, we will frequently use the discrete gradient operator $D : \mathbb{R}^n \to \mathbb{R}^{2 \times n}$ given as

$$Dx \coloneqq \begin{pmatrix} (D_h x)^\top \\ (D_v x)^\top \end{pmatrix}.$$

The discrete gradient operator essentially concatenates the horizontal and vertical image gradient vectors into a matrix with two rows.

4.2 Statistics of Natural Images

The pioneering study of neurons in the visual cortex conducted by Hubel and Wiesel [159, 160] led to a deeper understanding of the visual perceptual systems

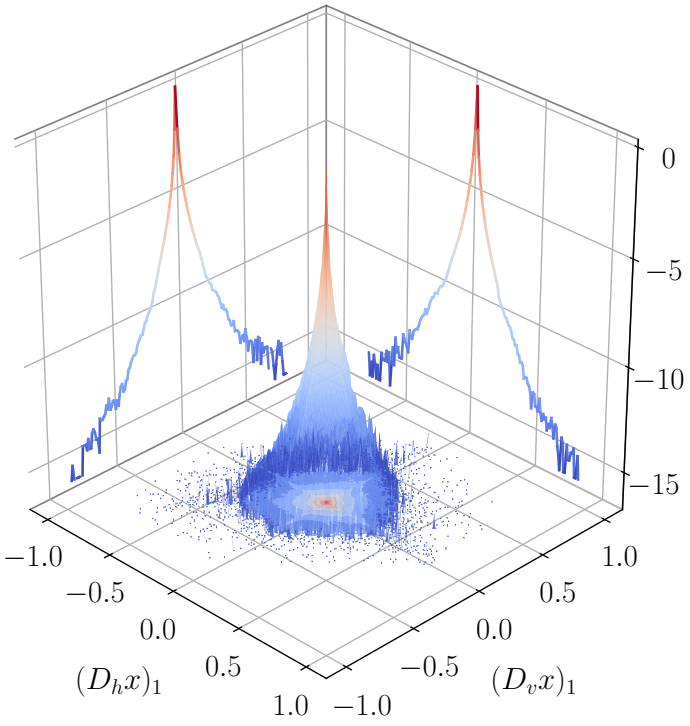


Figure 4.2: Visualization of the logarithmic joint density of the horizontal and vertical image gradient in the upper left corner $\log p((D_h x)_1, (D_v x)_1)$, estimated using $14,000,000$ patches of size 2×2 from the campus dataset [162]. The bottom-plane includes a contour plot of the estimated density. The left plane shows the estimated logarithmic marginal statistics of just the vertical gradients $\log p((D_v x)_1)$, while the right plane shows its horizontal equivalent $\log p((D_h x)_1)$. Cool colors indicate areas with low density, while warm colors mark areas with higher density.

of mammals and showed that the visual stream is processed by a sequence of transformations. Initially, Field [35] as well as Ruderman and Bialek [36] characterized natural images and studied their statistics to better understand the evolution of the mammalian visual system. Later, the computer vision community studied the statistics of natural images to find more realistic priors and thereby regularizers in order to improve the performance in various applications such as image reconstruction, denoising, and compression. Therefore, Huang and Mumford [43], Simoncelli [44], and Grenander and Srivastava [161] developed simple parametric probabilistic models that well describe the statistics of natural images in the gradient domain and the complex dependency of pairs of wavelet coefficients. Following their work, we empirically analyze the statistics of gradients and image patches in this section.

4.2.1 Statistics of Gradients

We consider discrete images defined on a regular pixel grid and extract statistics of the horizontal and vertical image gradients since they are invariant to intensity shifts and capture the local structure of images. To this end, we consider 2×2 image patches from a dataset of natural images. Throughout this chapter, we use the campus dataset [162], which consists of 90 images of size $4,284 \times 2,844$ each stored with a precision of 16bit, as a source for natural images that also include human-made structures. We convert the images to floating-point precision and rescale them by $2^{16} - 1$ such that all intensities of an image are in $[0, 1]$. Then, we randomly extract N patches of size 2×2 from all images of the campus dataset [162] and processed them in the following manner. Let $x_i \in \mathbb{R}^4$ represent a 2×2 image patch using the vector representation defined in Section 4.1.1 for $i = 1, \dots, N$. We denote by $D_h \in \mathbb{R}^{4 \times 4}$ the horizontal finite difference operator and by $D_v \in \mathbb{R}^{4 \times 4}$ its vertical equivalent defined in Section 4.1.2. We use $N = 14,000,000$ patches of the campus dataset [162] to empirically estimate the marginal density distribution of horizontal and vertical gradients in the upper left corner $p((D_h x)_1, (D_v x)_1)$ of the patches. The resulting estimated density is depicted in Figure 4.2 in a logarithmic domain. The surface plot of the estimated density indicates that the distribution has a high kurtosis, a sharp apex at $(D_h x)_1 = (D_v x)_1 = 0$, and a long exponential tail. This means that the majority of the image gradients is actually rather small, which reflects that images

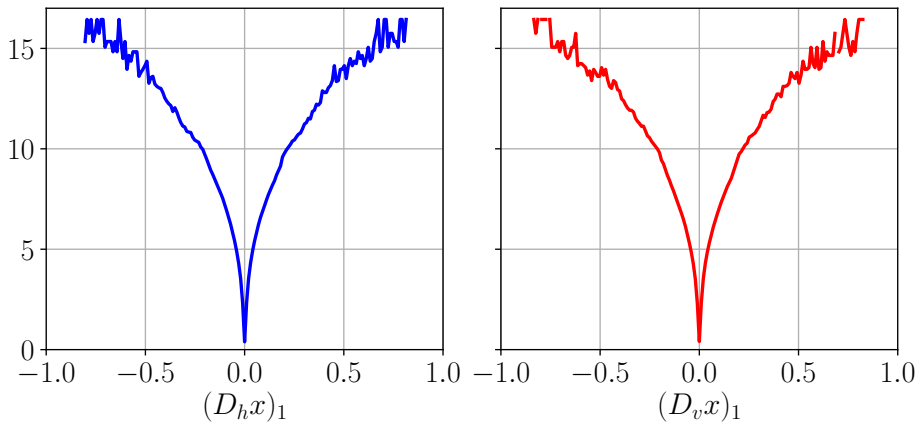


Figure 4.3: Estimated marginal distribution of the horizontal (left, $-\log p((D_h x)_1)$) and vertical (right, $-\log p((D_v x)_1)$) image gradients in a negative logarithmic domain.

typically consist of objects with a distinct color, e.g. sky and street, or textured regions with small color variations such as sand, grass, and soil. However, large gradients, corresponding to sharp edges between objects in an image, are typically very sparse. In addition, the contour plot at the bottom plane indicates that the density of the vertical and horizontal gradients is not independent.

In order to link the statistics of image gradients with regularizers, we plot the negative logarithm of the estimated marginal horizontal and vertical image gradient density in Figure 4.3. The marginal distributions of the horizontal and vertical gradients also have a high kurtosis, a sharp apex at zero, and a long exponential tail. Thus, a regularizer for image gradients should assign a penalization energy that reflects the strength of the gradient, since small gradients occur quite frequently while large gradients are sparse. As a result, a good regularizer for image gradients describes the statistics of natural images in the negative log-domain well. Further below in this chapter, we discuss classical principle-based regularizers, their mathematical properties, and how well they represent the statistics of natural images.

4.2.2 Statistics of 2×2 Image Patches

To analyze the local geometrical structure of 2×2 image patches, we follow the approach of Lee et al. [163] and compute image statistics on the zero-mean and normalized sphere in the discrete cosine transform (DCT) domain. In detail, let $x_i \in \mathbb{R}^{n^2}$ for $i = 1, \dots, N$ represent samples of $n \times n$ image patches as discussed in Section 4.1.1. We transform a vector representation of a patch x_i into the DCT domain by using the transformation matrix $B_{n\times n} = (b_{i,j})_{i,j=1}^{n^2} \in \mathbb{R}^{n^2 \times n^2}$, whose entries are given by

$$b_{ln+m+1,in+j+1} = 4\alpha(l)\alpha(m)\cos\left(\frac{\pi l(2i+1)}{2n}\right)\cos\left(\frac{\pi m(2j+1)}{2n}\right) \tag{4.1}$$

for $0 \leq l, m, i, j \leq n-1$ with the scaling coefficients

$$\alpha(i) = \begin{cases} \sqrt{\frac{1}{4n}} & \text{for } i = 0, \\ \sqrt{\frac{1}{2n}} & \text{for } 0 < i \leq n-1. \end{cases}$$

These entries are the coefficients of the inverse DCT-II transform [164] and the rows of the transformation matrix $B_{n\times n}$ hold the vector representations of the DCT filters forming an orthonormal basis. Here, we are interested in the statistics of zero-mean 2×2 image patches ($n = 2$). Thus, the corresponding transformation matrix is given

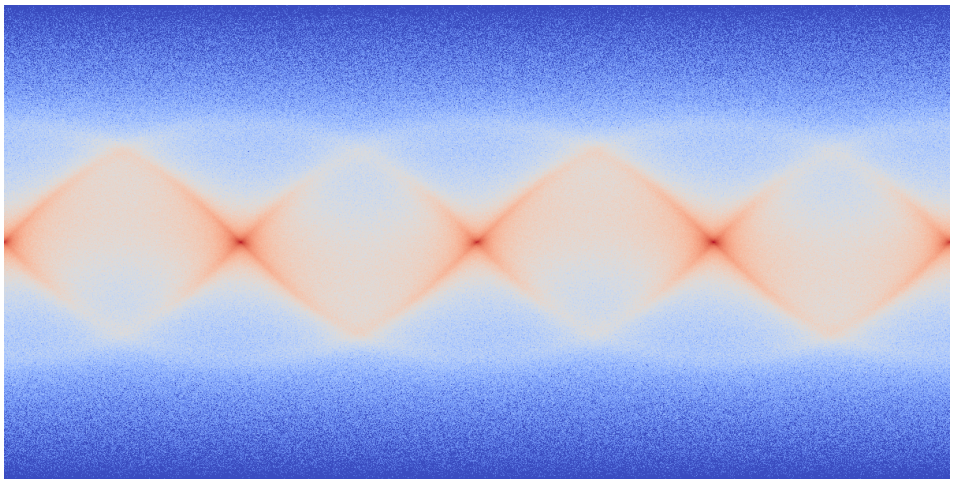


Figure 4.5: Illustration of the estimated density of zero-mean and normalized DCT coefficients of 2×2 image patches on the unit sphere $\mathcal{S}^2$ as a function of the lateral and longitudinal angles. The estimated density is color-coded using a logarithmic scale, where warmer colors indicate higher densities.

by

$$B_{2\times 2} = \frac{1}{2}\begin{pmatrix} 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix},$$

where we omitted the first row that computes the patch average. Then, the DCT coefficients $y_i \in \mathbb{R}^3$ are given by

$$y_i = B_{2\times 2} x_i$$

for $i = 1, \ldots, N$. Finally, we project the DCT coefficients onto the unit sphere $\mathcal{S}^2 := \left\{ y \in \mathbb{R}^3 : \left\| y \right\|_2 = 1 \right\}$ by normalizing the coefficients

$$\widehat{y} = \frac{y}{\left\| y \right\|_2}.$$

The resulting density on $\mathcal{S}^2$ of the same $14,000,000$ patch samples as in the previous section is depicted in Figure 4.4 using a logarithmic color code, where warmer colors indicate regions with higher density. To grasp the entire statistics of natural 2×2 image patches, we visualize the whole surface of the unit sphere $\mathcal{S}^2$ by parametrizing its surface by longitudinal and lateral angles in Figure 4.5. The resulting estimated density resembles a projection of a double helix with four distinct regions of maximal density at the equatorial regions and diminishing densities at polar regions. Note that the two poles correspond to the two possible checkerboard patches, while step edges with all possible orientations are located along the equator. Further, the density within the four rhombi is larger than outside and alternates from left to right. In the first rhombus on the left, the density inside is larger in the northern region, while in the second-left rhombus it is larger in the southern regions. This alternating pattern continues in the third and fourth rhombus. In addition, the top and the bottom corner of adjacent rhombi are connected by a small band that represents edge patches in different orientations and intensities. As a result, the statistics of natural 2×2 image patches clearly do not follow a Gaussian density distribution and the density does not factorize along the DCT coefficients.

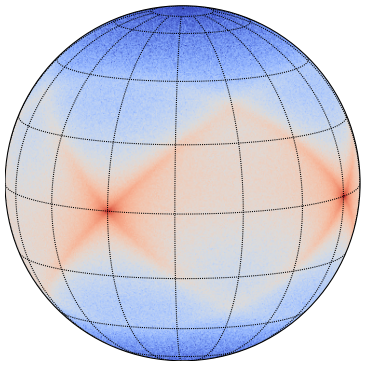


Figure 4.4: Illustration of the estimated density of zero-mean and normalized DCT coefficients of 2×2 image patches on the unit sphere $\mathcal{S}^2$. The estimated density is color-coded using a logarithmic scale, where warmer colors indicate higher densities.

4.2.3 Statistics of DCT and Random Filter Responses

The DCT filters and the thereby extracted coefficients are widely used in computer vision applications (e.g. JPEG compression [165]) and random filters are common in machine learning [166, 167]. Thus, analyzing the statistics of their responses for natural images is of high interest. To compute the DCT-II $n \times n$ basis filters, we first compute the $B_{n\times n} \in \mathbb{R}^{n^2 \times n^2}$ transformation matrix defined in Equation (4.1). Then,

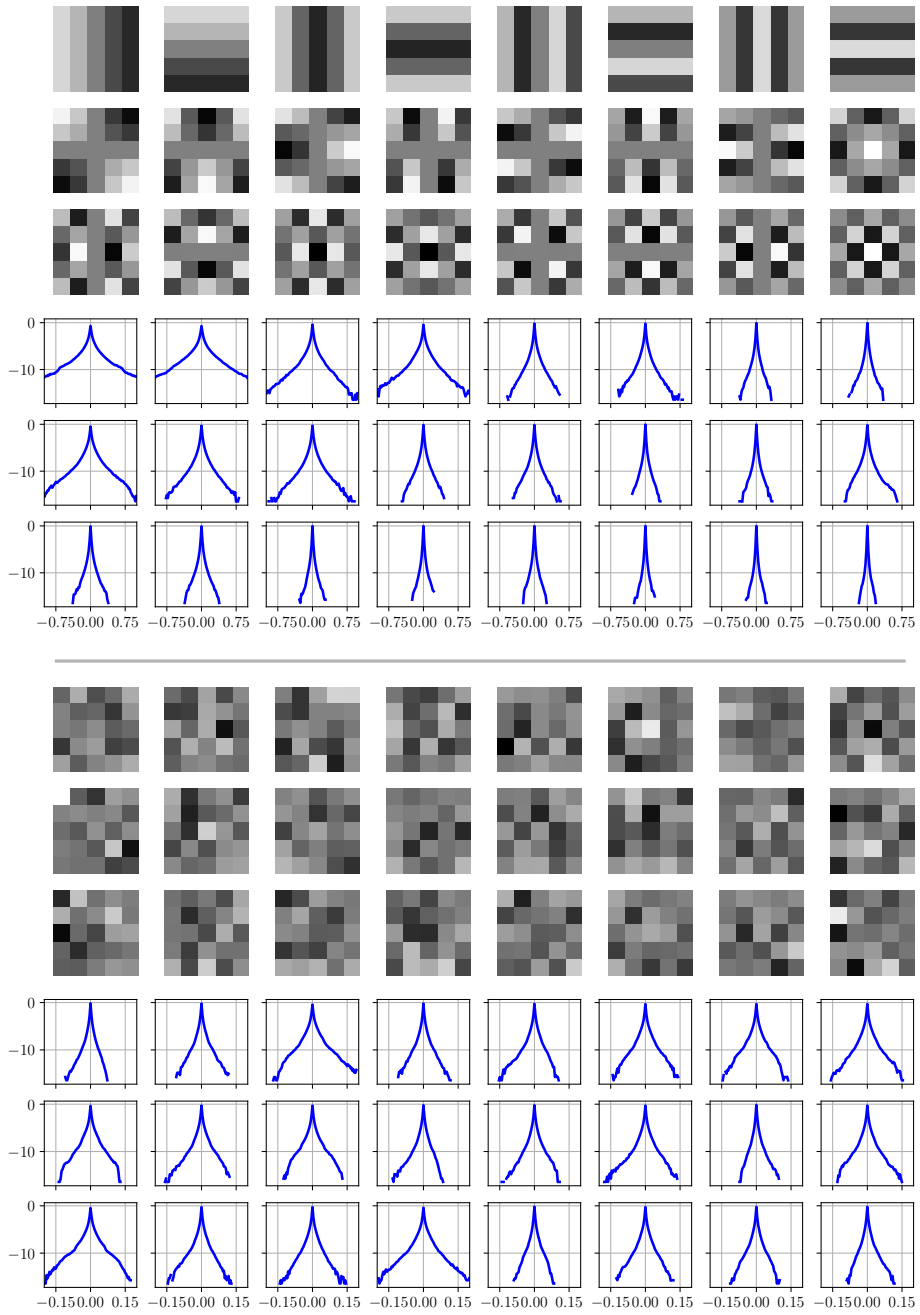


Figure 4.6: Statistics of responses of various filters applied to the considered natural image dataset. The top row depicts the 24 DCT filters of size 5×5 along with the estimated density using a logarithmic scale, while the bottom row depicts random filters, where each coefficient is drawn from a normal distribution $\mathcal{N}(0, 1)$, along with the estimated logarithmic density. Note that each filter has zero-mean and is normalized such that its ℓ^2-norm equals one. The density distributions have the same support in each row. For the DCT responses, the support is $[-1, 1]$, while we used $[-0.2, 0.2]$ for the random filters to account for the smaller amplitudes of the responses.

each row is the vector representation of an $n \times n$ DCT filter. The corresponding 2-dimensional filters for $n = 5$ are depicted in the top row in Figure 4.6. We omit the constant filter since it just reflects the local average in the $n \times n$ neighborhood of a pixel. The remaining basis filters all have zero-mean and are normalized such that their ℓ^2-norm equals 1. These filters are sorted such that the high-frequency components increase form top left to bottom right starting from low-frequency step edge filters in the horizontal and vertical direction. Using these basis filters, we compute the response of $14,000,000$ image patches of size 5×5 sampled from the campus dataset [162]. The estimated marginal densities across all considered patches are depicted in Figure 4.6 below the DCT basis filters. The statistics of the first basis filter responses are characterized by a high kurtosis, a sharp apex at zero, and a long exponential tail, just as the marginal statistics of horizontal and vertical image gradients, see Figure 4.3. The higher frequent the basis filters become, the smaller becomes the tail of the associated marginal distribution and its peak at zero turns sharper. However, the overall shape of the distribution remains unchanged. To sum up, the statistics of DCT basis filter responses indicate that there are few patches with

high-frequency components in natural images (smaller tails of the corresponding density distributions) and the shape of all estimated density distributions is similar, independent of the associated basis filter.

To ensure that the independence of the shape of the density distribution is a property of natural images and not due to the particular structure of the DCT-II basis filters, we evaluate the filter responses of random filters as suggested by Huang [168]. To this end, we randomly sample coefficients $\widehat{k}_{i,j} \sim \mathcal{N}(0,1)$ for $1 \leq i,j \leq n$ from a normal distribution and construct zero-mean and normalized filters $K \in \mathbb{R}^{n\times n}$ by setting their coefficients to

$$k_{i,j} = \frac{\widehat{k}_{i,j} - \mu}{\sqrt{\sum_{l,m=1}^{n} \left(\widehat{k}_{l,m} - \mu\right)^2}}$$

for $1 \leq i,j \leq n$ with

$$\mu = \sum_{i,j=1}^{n} \widehat{k}_{i,j}.$$

We show 24 of these random filters for $n = 5$ at the bottom in Figure 4.6 along with the estimated density distributions of the corresponding responses for the natural image patch dataset defined above. Clearly, the overall shape of the density distributions does not change and is independent of the corresponding filter. As a result, it is an intrinsic property of natural images that the logarithmic density distributions of coefficient extracted by zero-mean and normalized filters have a high kurtosis, a sharp peak at zero, and an exponential tail.

There are further aspects of the statistics of natural images such as their scale invariance [35, 36, 169] or self-similarity [35, 170, 171] that can be exploited to improve for instance image denoising [72]. However, for the sake of conciseness, we do not provide further details.

4.3 First-principle-based Regularizers

Inverse problems are often ill-posed in the sense of Hadamard [9], i.e., their solution is typically unstable with respect to perturbations of the observations (measurement noise), numerical errors in the computation of the solution, or there is a whole family of solutions that describe the observations equally well, e.g. accelerated MRI or undersampled CT. Originally, regularization techniques were developed to address these instabilities of ill-posed inverse problems and based on the works of Tikhonov [22, 172] a broad field of literature on regularization techniques evolved in the last six decades. In his original work Tikhonov considered linear inverse problems of the form $z = Ax \in \mathbb{R}^l$ with $A \in \mathbb{R}^{l\times n}$ and proposed to solve

$$\min_{x\in\mathbb{R}^n} \tfrac{1}{2} \|Ax - z\|_2^2 + \tfrac{\lambda}{2} \|x - x_0\|_2^2$$

for $\lambda > 0$ and an a-priori estimate of the solution $x_0 \in \mathbb{R}^n$ to end up with a stable solution $x \in \mathbb{R}^n$. Since we typically do not know the solution a-priori, we assume $x_0 = 0$. The solution to the resulting problem is given by

$$x = (A^*A + \lambda \,\mathrm{Id})^{-1} A^* z,$$

where A^* is the adjoint matrix as specified in Definition 2.1.12. For $\lambda > 0$ the inverse can be computed since $A^*A + \lambda \,\mathrm{Id}$ is positive definite and symmetric for any A. The stability of the solution with respect to perturbations of the observations then strongly depends on the condition number of $(A^*A + \lambda \,\mathrm{Id})^{-1}A^*$, which can be controlled by setting λ properly.

While in many machine learning settings the penalization of the ℓ^2-norm of the solution is beneficial (e.g. weight decay, support vector machines (SVMs)) [32, 34], we typically avoid penalizing the image intensities in imaging problems since it introduces a bias towards dark images with low intensities. Therefore, we follow the idea of Phillips [23] and incorporate a prior concerning the smoothness of the solution by penalizing the quadratic ℓ^2-norm of the image gradients. Then, the solution is given by

$$\min_{x \in \mathbb{R}^n} \tfrac{1}{2} \|Ax - z\|_2^2 + \tfrac{\lambda}{2} \|Dx\|_F^2 ,$$

which is unique if the intersection of the null space of both operators only includes the zero vector, i.e. $\ker(A) \cap \ker(D) = \{0\}$, because only then the resulting linear system of equations $(A^*A + \lambda D^*D)x = A^*z$ has a unique solution. Penalizing the intensity of the gradients is better suited for imaging problems since the majority of the gradient magnitudes in the horizontal and vertical direction are small as we have seen in the previous section.

To numerically check this assumption and compare different regularizers, we consider the Gaussian denoising problem throughout the next sections in this chapter. Let $y \in \mathbb{R}^n$ be the ground truth image depicted in Figure 4.7 and $z = y + \xi \in \mathbb{R}^n$ the observed image degraded by additive Gaussian noise $\xi \sim \mathcal{N}(0, \sigma^2 \operatorname{Id})$ with standard deviation $\sigma = 0.1$, as illustrated in Figure 4.8. The task is to compute an estimate $\widehat{x}$ that is as close as possible to the ground truth y by solving

$$\widehat{x} = \operatorname*{argmin}_{x \in \mathbb{R}} \tfrac{1}{2} \|x - z\|_2^2 + \mathrm{R}(x) \tag{4.2}$$

for different regularizers $\mathrm{R} : \mathbb{R}^n \to \mathbb{R}$.

As a first regularizer we consider the quadratic gradient penalization

$$\mathrm{R_Q}(x) = \tfrac{\lambda}{2} \|Dx\|_F^2$$

motivated by Phillips [23]. In the Gaussian image denoising case the linear operator is given by the identity operator ($A = \mathrm{Id}$), hence the solution of (4.2) is unique ($\ker(\mathrm{Id}) = \{0\}$) and given by

$$\widehat{x} = (\mathrm{Id} + \lambda D^*D)^{-1} z .$$

Figure 4.9 shows the restored image $\widehat{x}$ for $\lambda = 0.5$. We determined the value for λ by a grid search such that the PSNR score of the resulting solution is maximal. Although the PSNR score increased by more than 4dB compared to the noisy image, the quality of the image is not satisfactory. The image is still heavily degraded by noise and if λ was further increased, the noise along with the image edges would be smoothed out, which would result in low-frequency noise and blurry image edges.

Tikhonov's as well as Phillips' approach amount to computing solutions of inverse problems by solving a least squares problem, first defined by Carl Friedrich Gauss in 1795 and first published by Adrien-Marie Legendre in 1805 [173]. From a statistical point of view, the quadratic penalization of the gradient implies that a Gaussian distribution $\mathcal{N}(0, \frac{1}{\lambda})$ with variance $\frac{1}{\lambda}$ of the horizontal and vertical image gradient is assumed. However, the gradients of natural images are not Gaussian distributed as we have seen in the previous section. In summary, the quadratic penalization of image gradients can be solved efficiently in closed form but does not properly account for the statistics of natural images.

Figure 4.7: Original image of the water castle Château d'Azay-le-Rideau by Jean-Christophe Benoist (CC-BY-3.0).

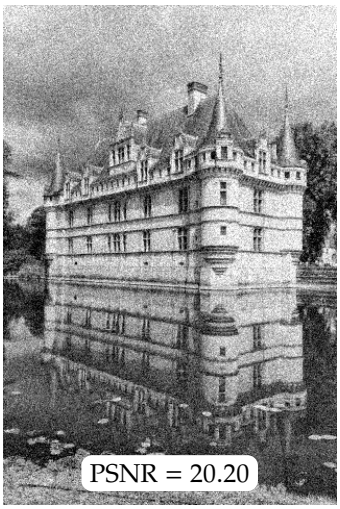


Figure 4.8: Noisy water castle image.

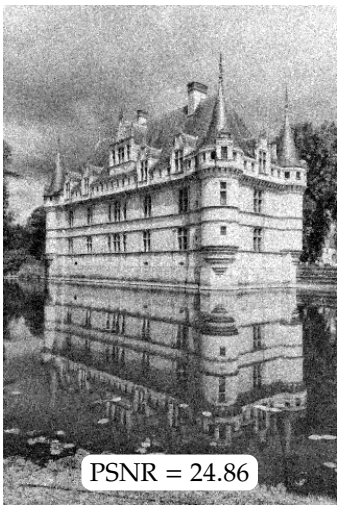


Figure 4.9: Denoised water castle image using quadratically penalized gradients.

4.3.1 Total Variation

Rudin pointed out in his PhD thesis [174] that the total variation (TV) norm, which essentially accounts for the absolute gradient jumps within an image, should be used

in imaging to preserve sharp discontinuities such as step edges that frequently appear between objects. Three decades after the publication of Tikhonov's work, Rudin, Osher, and Fatemi [1] applied the TV regularizer to Gaussian image denoising and paved the way for a new flourishing research direction. In this section, we discuss the TV regularizer and explore further TV-based regularizers.

The TV for discrete images $x \in \mathbb{R}^n$ considered in this thesis is given by

$$\mathrm{TV}(x) := \|Dx\|_{2,1}\,, \tag{4.3}$$

where D is the finite difference operator and the $\|\cdot\|_{2,1}$ computes the ℓ^1-norm of the ℓ^2-column norms. For an image $x \in \mathbb{R}^n$, the isotropic TV can be computed by

$$\mathrm{TV}(x) = \sum_{i=1}^{n} \sqrt{(D_h x)_i^2 + (D_v x)_i^2}.$$

Hence, it penalizes the absolute value of the gradient magnitude at each pixel. The corresponding TV regularizer is then defined as

$$\mathrm{R}_{\mathrm{TV}}(x) := \lambda \|Dx\|_{2,1}\,, \tag{4.4}$$

where $\lambda > 0$ is a parameter that defines the strength of the regularization. In contrast to the quadratic penalization of gradients, the TV regularizer implicitly assumes that the image gradients at each pixel follow a Laplace distribution $\mathrm{Laplace}(0, \frac{1}{\lambda})$ with the corresponding probability density function

$$p_{\mathrm{Laplace}}(x) = \frac{\lambda}{2} \exp\left(-\lambda \|Dx\|_{2,1}\right).$$

We depicted the associated Gaussian and Laplacian density functions in a negative logarithmic domain in Figure 4.10. Clearly, both the Gaussian and Laplacian distribution do not describe the statistics of natural images well. Nevertheless, the Laplacian distribution is a better model since it is closer to the empirically estimated distribution of natural images. In fact, the absolute function associated with the TV model is the best convex approximation of the estimated statistics. However, the better approximation of the real density of image gradients of the TV regularizer leads to the caveat that optimization is computationally more complex because the absolute function is not continuously differentiable. This non-differentiability was first addressed by smoothing the TV [27, 39, 175, 176] such that

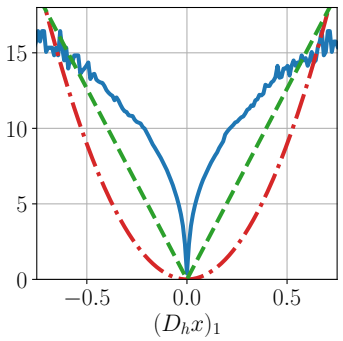


Figure 4.10: The solid blue curve shows the estimated negative logarithmic density of the horizontal gradients of natural images as discussed in Section 4.2.1, the dash-dotted red curve depicts $\frac{\lambda}{2}(D_h x)_1^2$ with $\lambda = 75$ and the dashed green curve depicts $\lambda|(D_h x)_1|$ with $\lambda = 50$. Note that both λ values were estimated in the least squares sense.

$$\mathrm{TV}_\beta(x) = \sum_{i=1}^{n} \sqrt{(D_h x)_i^2 + (D_v x)_i^2 + \beta^2}$$

is continuously differentiability for $\beta > 0$. Later, Fenchel's duality of TV

$$\mathrm{R}_{\mathrm{TV}}(x) = \max_{y \in \mathbb{R}^{2\times n} : \|y\|_{2,\infty} \le \lambda} \langle y,\, Dx \rangle\,,$$

is used [177–179] to dualize the TV-ℓ^2 model for image denoising, which results in the smooth dual problem. We show this by starting with the plain TV-ℓ^2 model for image denoising

$$\min_x \mathrm{R}_{\mathrm{TV}}(x) + \tfrac{1}{2}\|x - z\|_2^2\,.$$

Since the TV is a convex function, it is identical to its biconjugate, see Theorem 2.4.21. Thus, we have the equivalent problem

$$\min_x \max_{y \in \mathbb{R}^{2\times n} : \|y\|_{2,\infty} \le \lambda} \langle y,\, Dx \rangle + \tfrac{1}{2}\|x - z\|_2^2\,.$$

Under mild conditions (Neumann's minmax theorem [180]), we can swap the mini-

mum and maximum to get

$$\max_{y\in\mathbb{R}^{2\times n}:\|y\|_{2,\infty}\leq\lambda} \min_{x} \langle y, Dx\rangle + \tfrac{1}{2}\|x-z\|_2^2,$$

where the inner minimization problem is smooth in x with closed form solution

$$x = z - D^*y.$$

Plugging this result into the maximization problem yields the equivalent dual problem

$$\min_{y\in\left\{y\in\mathbb{R}^{2\times n}:\|y\|_{2,\infty}\leq\lambda\right\}} \tfrac{1}{2}\left\|D^*y\right\|_2^2 - \langle y, Dz\rangle, \tag{4.5}$$

which can be efficiently solved by for instance FISTA [99]. Building on the Fenchel duality of TV, Chambolle and Pock proposed the primal-dual hybrid gradient method [66] that does not require any smoothing, additionally accounts for a linear operator A and has an optimal convergence rate.

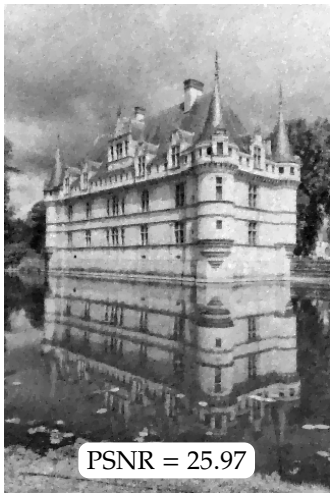


Figure 4.11: Denoised water castle image using the TV regularizer that penalizes the magnitude of image gradients.

If we apply the TV regularizer $\mathrm{R_{TV}}$ to the Gaussian denoising problem (4.2) of the water castle, we get the solution depicted in Figure 4.11. This solution was computed by solving the dual problem (4.5) with $\lambda = 0.06$ using FISTA [99], where λ was determined by a grid search such that the PSNR score of the resulting denoised image is maximal. Compared to the previous result obtained by $\mathrm{R_Q}$, the image is actually denoised and sharp edges are preserved and also the PSNR score increased by more than 1 decibel. However, homogeneous regions of the image such as the sky, walls or the water are not well restored and are in fact degraded by patchy step artifacts. This issue originates from the fact that TV favors piecewise constant solutions, which is known in the literature as the staircasing phenomenon of TV [27, 181].

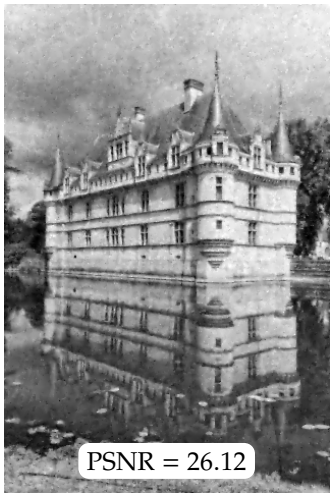


Figure 4.12: Denoised water castle image using the TGV^2 regularizer that does not penalize constant image gradients.

To overcome the staircasing problem, the first-principle assumption on the image gradients of TV has been extended to piecewise smooth images incorporating higher-order image derivatives [27, 29, 182]. Chambolle and Lions [27] proposed to address the staircasing effect by means of the infimal convolution problem

$$\min_{x_1+x_2=x} \|Dx_1\|_{2,1} + \left\|\widetilde{D}^2x_2\right\|_{2,1} + \|Ax-z\|_2^2,$$

where the additional term penalizes the TV of the image gradients and the operator $\widetilde{D}^2$ extracts the gradient of the image gradients. Later, Bredies, Kunisch, and Pock [29] introduced the total generalized variation (TGV) as an extension of TV to regularize and balance higher-order derivatives of images. TGV does not suffer from the staircasing effect. The second-order TGV explicitly allows affine image intensity profiles and is defined for discrete images as

$$R_{\mathrm{TGV}^2}(x,v) \coloneqq \alpha_1\|Dx-v\|_{2,1} + \alpha_0\left\|D^2v\right\|_{2,1},$$

where the operator $D^2 : \mathbb{R}^{2\times n} \to \mathbb{R}^{3\times n}$ computes the symmetrized gradients of the vector field $v \in \mathbb{R}^{2\times n}$ and $\alpha_1, \alpha_0 > 0$ are weights that balance the regularization strength of the different derivatives. The underlying idea of this regularizer is that the vector field v encodes the affine intensity profiles of the image such that they do not affect the first TV term.

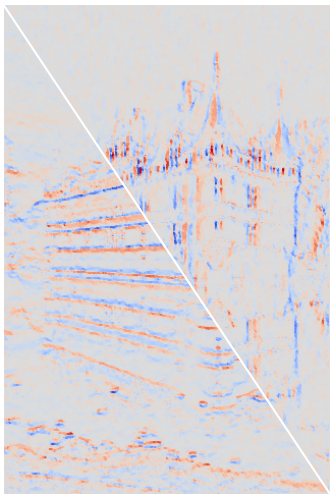


Figure 4.13: Horizontal (upper right) and vertical component (lower left) of the resulting TGV^2 vector field v displayed in the interval $[-\frac{1}{4}, \frac{1}{4}]$. Here, cool and warm colors represent negative and positive values, respectively.

Figure 4.12 depicts the computed image of the TGV^2 regularizer, where $\alpha_1 = 0.06$ and $\alpha_0 = 0.04$ are adopted to the considered Gaussian image denoising problem. To solve this problem, we used the primal-dual hybrid gradient algorithm [66] and as before determined the values for α_1 and α_0 by a grid search maximizing the PSNR score of the corresponding solution image. Compared to the result of the TV regularizer (Figure 4.11) the PSNR score increased marginally by 0.15dB and the staircasing

effect was reduced especially at the roof of the towers. In particular, the vector field v depicted in Figure 4.13 indicates where the TGV^2 regularizer accounts for the local affine image structures to avoid staircasing. Nevertheless, the homogeneous regions such as the sky or water are still degraded by patchy structures.

4.3.2 Non-convex Regularizers

In their seminal work Geman and Geman [33] link image restoration problems with finding low-energy states of a corresponding physical system. The energy function of the system defines an associated Gibbs distribution, which comes along with a Markov random field (MRF) [34] image model due to the Gibbs distribution and MRF equivalence. To account for the ill-posed nature of inverse problems, Geman and Geman imposed a-priori knowledge about the solution by regularizing image gradients. In MRFs penalization functions are typically called potential functions. As we have seen in the previous section, a quadratic potential function is not well suited for image restoration as it leads to over smooth solutions due to the underlying global smoothness assumption. However, real images are typically composed of smooth objects that are separated by sharp interfaces. Therefore, a more realistic model allows outliers in the gradients, which correspond to image edges. Motivated by robust statistics, a zoo of potential functions was proposed in the literature, which can be roughly separated into convex [37–39] and non-convex [6, 40–43, 183–185] potential functions. For an in-depth survey of different potential functions, we refer to [39, 186]. In this section, we elaborate on smooth and non-convex potential functions that explain the statistics of natural images well.

One of the first non-convex potential functions that allows for edge preservation is due to Geman and McClure [6] and defined as

$$\psi_{\mathrm{Ge}}(x) = \frac{\alpha x^2}{1 + (\beta x)^2},$$

where $\alpha, \beta > 0$ are parameters to adjust the shape of the potential function. Later, Hebert and Leahy [42] as well as Huang and Mumford [43] motivated the use of the potential function

$$\psi_t(x) = \alpha \log\left(1 + (\beta x)^2\right)$$

in imaging for shape-defining parameters $\alpha, \beta > 0$. This potential function assumes that the gradients of natural images follow a Student-t distribution with the associated probability density function

$$p_t(x) = \frac{1}{Z(\alpha, \beta)} \frac{1}{\left(1 + (\beta x)^2\right)^{\alpha}},$$

where $Z(\alpha, \beta)$ is a normalization factor. Figure 4.14 depicts both potential functions along with the estimated density function of horizontal gradients of the campus dataset [162]. Compared to the quadratic and absolute potential functions (see Figure 4.10), both potential functions fit the gradient statistics better and the Student-t potential function ψ_t is the best fit. Thus, a suitable regularizer based on the Student-t potential function is given by

$$\mathrm{R}_t(x) = \sum_{i=1}^{n} \sum_{d \in \{h,v\}} \alpha \log\left(1 + \beta^2 (D_d x)_i^2\right), \tag{4.6}$$

which equally penalizes the horizontal and vertical image gradients.

Let us again consider the Gaussian denoising problem of the water castle to evaluate the quality of R_t. To optimize the adapted smooth non-convex problem, we use the accelerated gradient method [98] with momentum $\frac{1}{\sqrt{2}}$, which helps to escape spurious

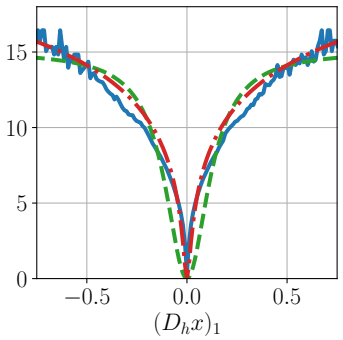


Figure 4.14: The solid blue curve shows the estimated negative logarithmic density of the horizontal gradients of natural images as discussed in Section 4.2.1, the dashed green curve depicts $\frac{\alpha (D_h x)_1^2}{1+(\beta (D_h x)_1)^2}$ with $\alpha = 878.38$, $\beta = 7.64$, and the dash-dotted red curve depicts $\alpha \log\left(1 + (\beta (D_h x)_1)^2\right)$ with $\alpha = 3.84$, $\beta = 78.85$. Note that in both cases α and β were estimated in the least squares sense.

local minima. Figure 4.15 depicts the resulting image, where the parameters α, β were determined by a grid search such that the PSNR score of the associated solution is maximized. The PSNR score has only marginally improved compared to the TV solution (see Figure 4.11). The solution associated with R_t is not degraded by the staircasing effect but homogeneous regions are degraded by low-frequency noise. Since this potential function describes the statistics of natural image gradients quite well (see Figure 4.14), a straight forward approach to improve the performance is the consideration of higher-order information of images. However, with increasing number of higher-order information, identifying these parameters manually becomes harder due to the curse of dimensionality. Therefore, a straight forward approach is to consider higher-order parametric regularizers and identify its parameters by means of learning [45, 46, 51, 53, 56, 57, 60, 61, 63, 187–189].

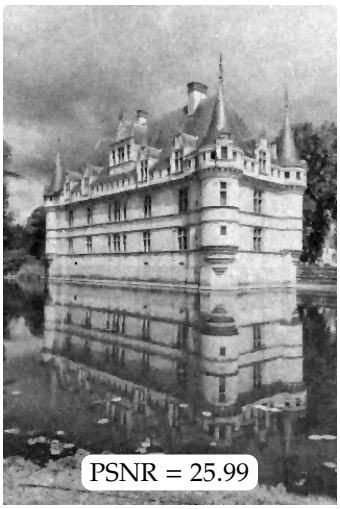


Figure 4.15: Denoised water castle image using the Student-t potential function on the image gradients.

We conclude this section with some remarks on the optimization of these non-convex energies. Geman and Geman [33] computed the solutions of the corresponding MRFs using stochastic methods such as simulated annealing [190], which avoid local minima by random permutations. This ensures convergence to global minima in non-convex optimization problems, however the computational effort is very high [33]. Later, Blake and Zisserman [41] proposed a deterministic algorithm called graduated non-convexity, where the non-convex potential function is approximated by a parametric function. Initially, this parametric approximation yields a convex potential function and during the optimization, the parametric function is adapted such that it eventually resembles the considered non-convex potential function. Nowadays, classical optimization algorithms such as accelerated gradient method, nonlinear conjugate gradient [191], or L-BFGS are used to solve these non-convex problems [52, 53]. However, these only guarantee convergence to stationary points.

4.4 Parametric Regularizers

In the previous section, we saw that regularizers operating only on the first-order statistics of natural images do not yield satisfactory results when used to denoise complex structured images such as the water castle image. In addition, the results were improved by penalizing second-order information using the TGV regularizer. Thus, an obvious idea is to consider higher-order statistics of images. However, modeling these higher-order statistics is challenging due to the large dimensionality of images and the complex statistics of large pixel neighborhoods [43, 44]. Therefore, parametric regularizers were developed whose parameters can be identified by learning from data.

To simplify the learning process and account for properties of natural images, it is reasonable to incorporate invariances in the parametric regularizer. Images capture arbitrary scenes of the universe, thus spatial translation invariance should be reflected in the regularizer design since objects are equally likely to appear at any image location. Furthermore, diverse lighting conditions manifest in various shadings of the same objects in different images. Hence, it is sensible to also consider radiometric shift-invariance in the design of a regularizer.

One of the first approaches to learn parameters of higher-order regularizers is due to Zhu and Mumford [45, 46]. They proposed a regularizer that penalizes images by applying potential functions $\lambda_j : \mathbb{R} \to \mathbb{R}$ to image features extracted by convolutional filters represented by the linear matrix operator $K_j \in \mathbb{R}^{n \times n}$ with the particular structure

$$R_{ZM}(x) := \sum_{i=1}^{n} \sum_{j=1}^{m} \lambda_j((K_j x)_i).$$

In their regularizer, they proposed to include horizontal and vertical derivative, Laplacian, and Gabor filter candidates at different scales and advocated potential

functions λ_j using piecewise constant interpolation on predefined bins. The interpolation weights of each potential function are determined such that it reproduces the statistics of the corresponding image feature. A filter pursuit algorithm determines whether the corresponding feature is included in the final regularizer. Due to the use of piecewise constant potential functions, this regularizer cannot be advocated in a first-order optimization-based image reconstruction approach.

A bit later Hinton [47] proposed to model high-dimensional data distributions by a product of experts f_j for $j = 1, \dots, m$, where each expert function can specialize on a distinct feature of the data, i.e.

$$p(x|\theta_1, \dots, \theta_m) = \frac{1}{Z} \prod_{j=1}^{m} f_j(x|\theta_j).$$

Here, θ_j for $j = 1, \dots, m$ are the parameters of the individual experts and Z is a normalization constant that ensures proper scaling of $p(x|\theta_1, \dots, \theta_m)$. Shortly afterwards, the product of experts model was applied to imaging [49] by modeling the statistics of $s \times s$ natural image patches $x \in \mathbb{R}^{s^2}$ by a family of experts, each following the Student-t model

$$f_j(x|\theta_j) = \frac{1}{\left(1 + \left\langle k_j, x \right\rangle^2\right)^{\alpha_j}},$$

where k_j is the j^{th} row of the linear matrix operator $K \in \mathbb{R}^{m \times s^2}$ and $\theta_j = (\alpha_j, k_j)$ denotes all parameters of the j^{th} expert, which are the weights $\alpha_j \in \mathbb{R}$ and the coefficients $k_j \in \mathbb{R}^{s^2}$. Note that $m > s^2$ enables an over-complete representation of the patch statistics. The parameters θ_j were estimated using contrastive divergence [48], which we discuss later in Section 4.4.3.1. In order to apply this patch-based prior model to full images, every overlapping patch of an image must be processed separately and the resulting image is given by a pixel-wise average of the overlapping patches. However, this approach is not suitable for most inverse problems since they typically do not decouple along image patches.

4.4.1 Fields of Experts

Roth and Black [51] introduced the fields of experts (FoE) model to learn generic image priors capturing the statistics of natural images by extending the product of experts model [47] to convolutional filters. Their resulting prior model for an image $x \in \mathbb{R}^n$ is given by

$$p_{\text{FoE}}(x, \theta) = \frac{1}{Z(\theta)} \prod_{i=1}^{n} \prod_{j=1}^{m} \exp\left(-\phi((K_j x)_i, w_j)\right),$$

where each $K_j \in \mathbb{R}^{n \times n}$ is a linear operator corresponding to a convolution filter of size $s \times s$ with coefficients $k_j \in \mathbb{R}^{s^2}$ and the potential functions $\phi : \mathbb{R} \times \mathbb{R}^{N_w} \to \mathbb{R}$ are parameterized by the weights $w_j \in \mathbb{R}^{N_w}$. In particular, this prior is translation invariant. All model parameters are combined in $\theta = (k_j, w_j)_{j=1}^m \in \Theta$, where $\Theta \subset \mathbb{R}^{m \times s^2 \times N_w}$ defines the space of feasible parameters. The associated FoE regularizer is then defined as the sum of the pixel-wise energy $\mathrm{r}_{\text{FoE}} : \mathbb{R}^n \times \Theta \to \mathbb{R}^n$,

$$\mathrm{R}_{\text{FoE}}(x, \theta) = \sum_{i=1}^{n} \mathrm{r}_{\text{FoE}}(x, \theta)_i = \sum_{i=1}^{n} \Psi(Kx, W)_i \tag{4.7}$$

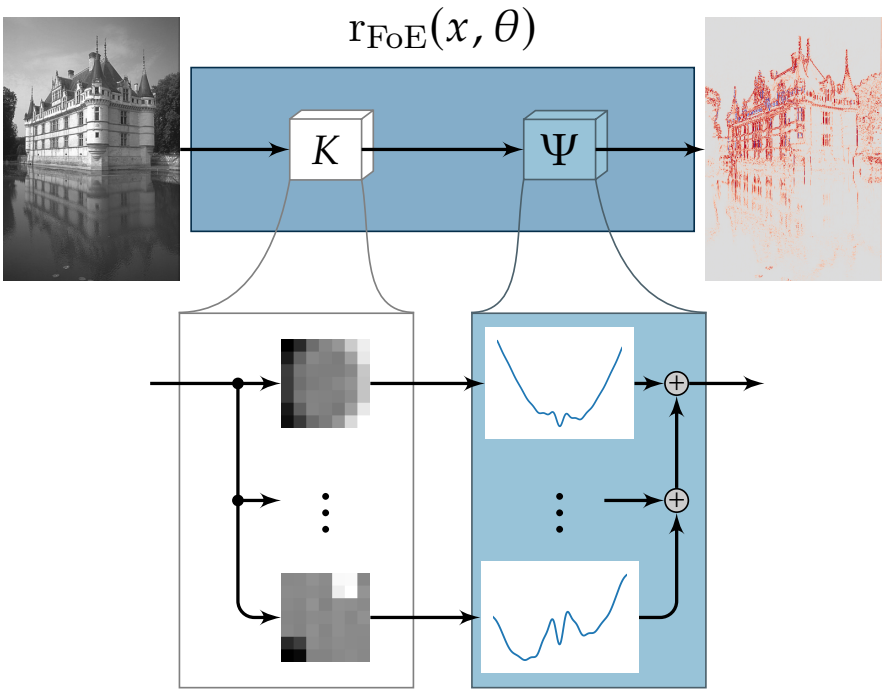


Figure 4.16: Visualization of the fields of experts regularizer. On the left an input image is depicted and the resulting pixel-wise energy r_{FoE} is depicted on the right, which is computed by extracting local features using convolution kernels and associated potential functions.

using the linear filter operator $K = (K_1 \ \dots \ K_m)^\top \in \mathbb{R}^{nm \times n}$ and the aggregated potential function $\Psi : \mathbb{R}^{nm} \times \mathbb{R}^{m \times N_w} \to \mathbb{R}^n$

$$\Psi(Kx, W)_i = \sum_{j=1}^{m} \phi((K_j x)_i, w_j),$$

where $W = (w_1 \ \dots \ w_m)^\top$ is a matrix aggregating the parameters of all potential functions. The computational structure of the FoE regularizer is visualized in Figure 4.16.

Roth and Black [51] considered the convex potential function $\phi(x, w) = w\sqrt{1 + x^2}$, which is a smooth approximation of the absolute function due to Charbonnier [39] as well as the non-convex potential function $\phi(x, w) = w \log(1 + x^2)$. In both cases, the potential functions are parameterized by a scalar weight $w \in \mathbb{R}$, which implies $N_w = 1$. To further increase the expressivity of the regularizer, Chen et. al [55] parameterized each potential function by weighted radial basis functions motivated by [192]. In the case of Gaussian radial basis functions, the altered potential function reads as

$$\phi(x, w) = \sum_{l=1}^{N_w} w_l \varphi\left(\frac{x - \mu_l}{\sigma}\right)$$

for a weight vector $w \in \mathbb{R}^l$, $\varphi(x) = \exp(-x^2)$, N_w equidistantly placed centers μ_l in the interval $[-v, v]$ and $\sigma = \frac{2v}{N_w - 1}$. Note that in this case, the interval specified by $v > 0$ should be large enough to capture the range of the filter responses. While in [51] the norm of a filter determines the steepness of the Student-t distribution, the flexibility of radial basis functions enabled [55] to fix the norm of the filter coefficients ($\left\|k_j\right\|_2 = 1$) and determine the maximal filter response for bounded images, which defines the extent v. An additional zero-mean constraint on the filter coefficients ensures that the filter response is approximately symmetric around zero and guarantees invariance against intensity shifts.

The receptive field of the pixel-wise FoE energy r_{FoE} is limited by the filter size s. Moreover, to express a certain image feature a specific expert filter has to specialize in detecting it since a convolution is essentially a correlation with a flipped filter. To encounter these issues, it is reasonable to define an image regularizer by means of deep learning principles, which enables a nonlinear combination of image features to

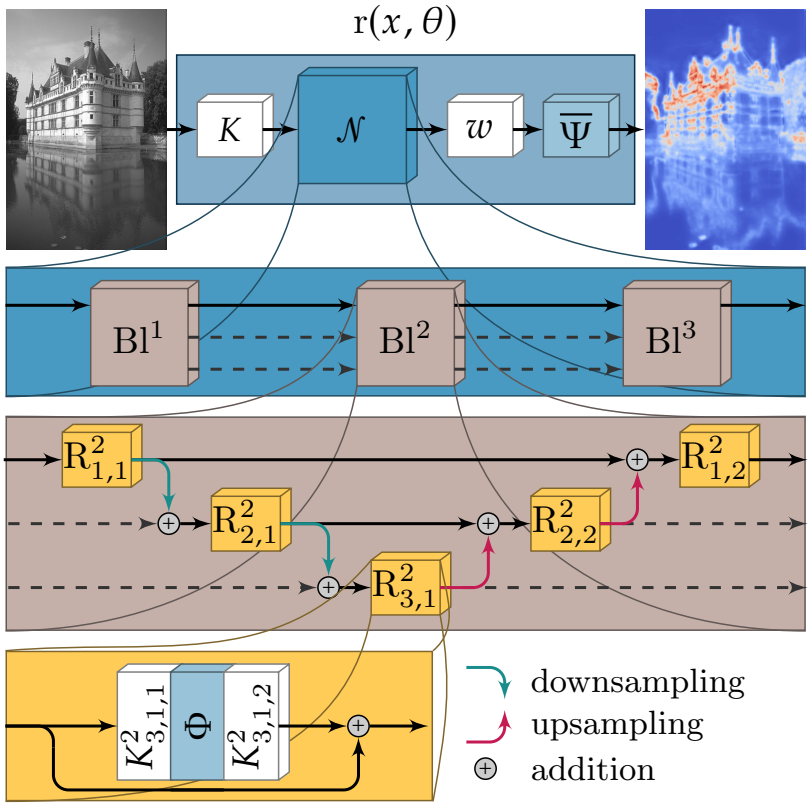


Figure 4.17: Visualization of the total deep variation regularizer (TDV^3_3). On the highest level, the deep variation $r(x,\theta) = \Psi(\mathcal{N}(Kx))$ assigns to each pixel an energy value incorporating the local neighborhood. The nonlinear function $\mathcal{N}$ (blue) is composed of three macro-blocks (gray), each representing a CNN with a U-Net-type architecture. Each macro-blocks consist of five micro-blocks (blue) with a residual structure on three scales.

build a richer local image regularizer with a larger spatial support (receptive field).

4.4.2 Total Deep Variation

To combine the mathematically well understood variational formulation with today's top-performing deep learning approaches, we proposed in [59, 60] the data-driven general-purpose total deep variation (TDV) regularizer. In contrast to many deep learning approaches that learn a direct mapping from a corrupted input image to the restored output [142, 193–195], the TDV regularizer is a deep convolutional neural network (CNN) that operates on multiple scales in successive blocks and assigns to each pixel in an image a local energy $\mathrm{r}(x,\theta)$. This regularization energy can be used in the variational formulation of inverse problems to incorporate prior knowledge in the reconstruction process.

The TDV regularizer for an image $x \in \mathbb{R}^{nC}$ of size $n = n_v \times n_h$ with C channels is defined as

$$R_{\mathrm{TDV}}(x,\theta) = \sum_{i=1}^{n} \mathrm{r}(x,\theta)_i = \sum_{i=1}^{n} \overline{\Psi}(w\mathcal{N}(Kx))_i, \tag{4.8}$$

where $K \in \mathbb{R}^{nm\times nC}$ is a linear operator that stacks m convolution operators $K_j \in \mathbb{R}^{n\times nC}$ with filters of support 3×3. The mapping $\mathcal{N} : \mathbb{R}^{nm} \to \mathbb{R}^{nm}$ is a multi-scale CNN and $w \in \mathbb{R}^{n\times nm}$ is a learned 1×1 convolution kernel. The nonlinear function $\overline{\Psi} : \mathbb{R}^n \to \mathbb{R}^n$ assigns an energy to every pixel and is defined as

$$(x_1,\dots,x_n) \mapsto (\psi(x_1),\dots,\psi(x_n)),$$

where $\psi \in C^2(\mathbb{R},\mathbb{R})$ is a *potential function*. We denote by θ the entity of learnable parameters, i.e. K, all parameters of $\mathcal{N}$ and w. The computational structure of the TDV regularizer is visualized in Figure 4.17. In detail, TDV^b_a for integers $a, b \geq 1$ consists of b blocks $\mathrm{Bl}^1,\dots,\mathrm{Bl}^b$ (gray blocks in Figure 4.17), each of them has a U-Net [196] type architecture, where on all a scales residual blocks $\mathrm{R}^i_{1,1}, \mathrm{R}^i_{1,2}, \dots, \mathrm{R}^i_{a-1,1}, \mathrm{R}^i_{a-1,2}, \mathrm{R}^i_{a,1}$

(yellow blocks in Figure 4.17) are applied. To increase the expressiveness of the network, residual connections are added between scales of consecutive blocks whenever possible. Each residual block $\mathrm{R}^i_{j,k}$ with $i \in \{1,\dots,b\}$, $j \in \{1,\dots,a\}$ and $k \in \{1,2\}$ exhibits the particular structure $\mathrm{R}^i_{j,k}(x,\theta) = x + K^i_{j,k,2}\Phi(K^i_{j,k,1}x)$ for convolution operators $K^i_{j,k,1}, K^i_{j,k,2} \in \mathbb{R}^{nm\times nm}$ of size 3×3 with m feature channels and no bias. Following [43], the Student-t potential is a suitable model for the statistics of natural images, that is why we choose the particular activation function $\Phi : \mathbb{R}^{nm} \to \mathbb{R}^{nm}$

$$(x_1,\dots,x_{nm}) \mapsto (\phi(x_1),\dots,\phi(x_{nm}))$$

using the component-wise function $\phi(x) = \frac{1}{2}\log(1+x^2)$ with the properties $\phi'(0) = 0$ and $\phi''(0) = 1$. Taking into account the work by Zhang [197], we use 3×3 convolutions and transposed convolutions with stride 2 for downsampling and upsampling in conjunction with a blur kernel to avoid aliasing.

Comparing the TDV (4.8) to the FoE (4.7) regularizer, we see that the TDV regularizer allows for an additional nonlinear transformation of the features before applying the potential function. Moreover, the inherent multi-scale architecture of the TDV regularizer enables an efficient extraction of higher-order image features defined over a large local neighborhood.

4.4.3 Parameter Identification

Before a parametric regularizer can be applied to a problem, its parameters θ need to be identified. While determining a suitable regularization weight of the TV regularizer in the TV-ℓ^2 model can be done manually or via grid search, the curse of dimensionality prevents this strategy if hundreds or thousands of parameters have to be identified, as in the case of the FoE or TDV regularizer. Therefore, we review various approaches that have been proposed in the literature to learn the parameters of regularizers in this section.

4.4.3.1 Contrastive Divergence

A first approach to learn the parameters $\theta \in \Theta$ of a parametric regularizer is to maximize its likelihood. Let $(\mathbb{R}^n, \mathfrak{F}_y, \mathbb{P}_y)$ be a complete probability space for natural images of size $n = n_v \times n_h$ with σ-algebra $\mathfrak{F}_y$ and probability measure $\mathbb{P}_y$. We denote by $y \in \mathbb{R}^n$ a random sample from the distribution of natural images $\mathcal{T}_y$. The associated probability density of the parametric regularizer R for this random sample reads as

$$\frac{\exp(-\mathrm{R}(y,\theta))}{\int_{\mathbb{R}^n}\exp(-\mathrm{R}(x,\theta))\,\mathrm{d}x}.$$

Then, the maximal likelihood estimator of the parameters $\theta \in \Theta$ over natural images is equivalent to minimizing the expected negative log-likelihood:

$$\min_{\theta\in\Theta}\left\{\mathrm{J}(\theta) \coloneqq \mathbb{E}_{y\sim\mathcal{T}_y}\left[R(y,\theta)\right] + \log\left(\int_{\mathbb{R}^n}\exp(-\mathrm{R}(x,\theta))\,\mathrm{d}x\right)\right\}.$$

For suitably smooth regularizers [87, Section 6.3, Theorem 6.28], the gradient of the expected negative log-likelihood w.r.t. θ is given by

$$\begin{aligned}\frac{\partial \mathrm{J}(\theta)}{\partial\theta} &= \mathbb{E}_{y\sim\mathcal{T}_y}\left[\frac{\partial \mathrm{R}(y,\theta)}{\partial\theta}\right] - \int_{\mathbb{R}^n}\underbrace{\frac{\exp(-\mathrm{R}(x,\theta))}{\int_{\mathbb{R}^n}\exp(-\mathrm{R}(\widehat{x},\theta))\,\mathrm{d}\widehat{x}}}_{\mathbb{P}_{\mathcal{M}}(x,\theta)}\frac{\partial \mathrm{R}(x,\theta)}{\partial\theta}\,\mathrm{d}x\\ &= \mathbb{E}_{y\sim\mathcal{T}_y}\left[\frac{\partial \mathrm{R}(y,\theta)}{\partial\theta}\right] - \mathbb{E}_{x\sim\mathcal{T}_{\mathcal{M}}}\left[\frac{\partial \mathrm{R}(x,\theta)}{\partial\theta}\right],\end{aligned} \tag{4.9}$$

where the second expectation is over the distribution $\mathcal{T}_{\mathcal{M}}$ associated with the induced probability space of the model $(\mathbb{R}^n, \mathfrak{F}_{\mathcal{M}}, \mathbb{P}_{\mathcal{M}})$. Thus, the gradient is given by the difference of the expected gradient of natural images and the expected gradient of model samples. The caveat of this maximum likelihood formulation is that the second expectation requires sampling from the model distribution, which is typically computationally demanding using for instance Markov chain Monte Carlo Gibbs sampling. Moreover, samples from the model distribution have in general high variance as pointed out by Hinton [48], which frequently does not lead to meaningful gradients. Therefore, Hinton proposed to approximate the gradient by contrastive divergence

$$\frac{\partial \mathrm{J}(\theta)}{\partial \theta} \approx \mathbb{E}_{y \sim \mathcal{T}_y} \left[\frac{\partial \mathrm{R}(y, \theta)}{\partial \theta} \right] - \mathbb{E}_{x \sim \mathcal{T}_{\mathcal{M}^1}} \left[\frac{\partial \mathrm{R}(x, \theta)}{\partial \theta} \right],$$

where the second expectation is computed over the distribution $\mathcal{T}_{\mathcal{M}^1}$ defined by a single Gibbs sampling step on the model distribution originating from the data distribution $\mathcal{T}_y$. This yields a very efficient approximation of the gradient with little computational overhead. The resulting gradient can be used by any suitable first-order method in order to maximize the expected log-likelihood.

Recently, Lunz et al. [61] advocated learning adversarial regularizers by minimizing the difference of the expected regularization energy over ground truth samples and corrupted samples. Note that the gradient of their objective function w.r.t. the model parameters has a similar structure as (4.9). The only difference is that in the second term the expectation is computed over the distribution of corrupted samples instead of the model distribution.

4.4.3.2 Bilevel Learning

Bilevel learning is a supervised learning approach to learn the parameters θ of a regularizer in a discriminative way [52, 53, 198], as discussed in Section 3.1.3. Let $(\mathcal{Y} \times \Xi, \mathfrak{F}, \mathbb{P})$ be a complete probability space on $\mathcal{Y} \times \Xi$ with σ-algebra $\mathfrak{F}$ and probability measure $\mathbb{P}$. We denote by (y, ξ) a pair of independent random variables modeling the data representing the ground truth image $y \in \mathcal{Y} \subset \mathbb{R}^n$ and additive noise $\xi \in \Xi \subset \mathbb{R}^l$ with associated distribution denoted by $\mathcal{T} = \mathcal{T}_{\mathcal{Y}} \times \mathcal{T}_{\Xi}$. Each ground truth image y represents an image of size $n = n_v \times n_h$ and is related to the additive noise ξ by means of the observation

$$z = Ay + \xi,$$

where $A \in \mathbb{R}^{l \times n}$ is the fixed task-dependent linear operator of this linear inverse problem. Then, bilevel optimization amounts to minimizing the distance of the ground truth and the restored output defined by the minimizer of the lower-level problem, i.e.

$$\min_{\theta} \mathbb{E}_{(y,\xi)\sim\mathcal{T}} \ell(x^*(y, \xi) - y) \quad \text{s.t.} \quad x^*(y, \xi) = \operatorname*{argmin}_x \mathrm{R}(x, \theta) + \tfrac{1}{2} \|Ax - z\|_2^2, \quad (4.10)$$

where ℓ is a differentiable loss function and R has to be twice continuously differentiable. An equivalent optimization problem incorporating the first-order condition of the lower-level problem reads as

$$\min_{\theta} \mathbb{E}_{(y,\xi)\sim\mathcal{T}} \ell(x(y, \xi) - y) \quad \text{s.t.} \quad D_1 \mathrm{R}(x(y, \xi), \theta) + A^*(Ax(y, \xi) - z) = 0.$$

To account for the constraint, we introduce Lagrange multipliers $p \in L^2(\mathcal{Y} \times \Xi, \mathbb{R}^n)$ and define the Lagrangian

$$\mathrm{L}(x, \theta, p) = \mathbb{E}_{(y,\xi)\sim\mathcal{T}} \ell(x(y, \xi) - y) + \left\langle D_1 \mathrm{R}(x(y, \xi), \theta) + A^*(Ax(y, \xi) - z), p(y, \xi) \right\rangle$$

for $x \in L^2(\mathcal{Y} \times \Xi, \mathbb{R}^n)$. A strategy to compute the gradient of the Lagrangian w.r.t. the model parameters by means of implicit differentiation is given by:

1. Draw a sample pair $(y, \xi) \sim \mathcal{T}$.
2. Solve the lower-level problem with high precision for each sample such that

$$\frac{\partial \mathrm{L}}{\partial p(y,\xi)} = D_1 \mathrm{R}(x(y,\xi),\theta) + A^*(Ax(y,\xi) - z) = 0.$$

3. Then, compute for each sample the Lagrange multipliers $p(y, \xi)$ by solving the equation

$$\frac{\partial \mathrm{L}}{\partial x(y,\xi)} = \frac{\partial \ell(x(y,\xi) - y)}{\partial x(y,\xi)} + \left(D_1^2 \mathrm{R}(x,\theta) + A^*A\right) p(y,\xi) = 0.$$

Finally, the gradient of the Lagrangian w.r.t. the parameters of the regularizer is given by

$$\frac{\partial \mathrm{L}}{\partial \theta} = \mathbb{E}_{(y,\xi)\sim\mathcal{T}} \frac{\partial}{\partial \theta} D_1 \mathrm{R}(x(y,\xi),\theta) p(x(y,\xi),\theta).$$

As before, this gradient can be used by any first-order method to estimate the parameters. There also exists a second-order approach to bilevel learning due to Kunisch and Pock [198] that is based on Newton's method.

While contrastive divergence learns the parameters of the regularizer in an unsupervised fashion by maximizing the likelihood over target images, bilevel learning is a supervised learning method that adapts the parameters of the regularizer such that the MAP estimator is close to the associated target. To compare these two approaches, let us recall the 2×2 image patch statistics discussed in Section 4.2.2. Contrastive divergence would learn the regularizer parameters θ such that $\exp(-\mathrm{R}(\,\cdot\,,\theta))$ approximates the prior density depicted in the first row of Figure 4.18. In contrast, bilevel learning amounts to adapting the parameters θ of the regularizer such that the posterior density is maximal at the target, as shown in the last row of Figure 4.18. Note that the posterior can be computed by multiplying the prior density with the likelihood density and normalization according to Bayes' Theorem 2.2.7. In this particular example, we assumed a Gaussian denoising problem such that $A = \mathrm{Id}$ in Equation (4.10), which results in the likelihood depicted in the middle row of Figure 4.18. To sum up, the discriminative learning flavor of bilevel learning allows to identify regularizer's parameters for a specific task and leads to superior results compared to generative learning approaches such as contrastive divergence [52, 53]. However, it requires to solve the lower-level problem with typically high precision.

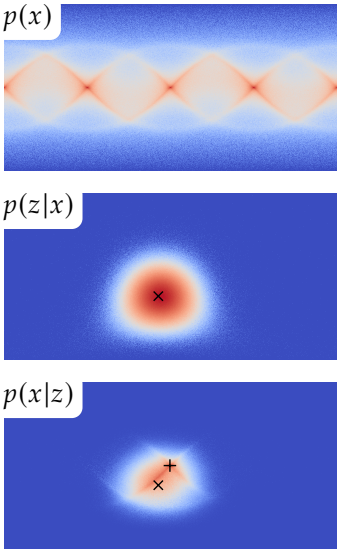


Figure 4.18: The top image depicts the prior density $p(x)$ of 2×2 image patches, as discussed in Section 4.2.2. The middle image illustrates the likelihood density $p(z|x)$ for a noisy patch z marked by the black x. The bottom image shows the associated posterior $p(x|z)$, where z is again marked by the black x, and the target y is marked by the black +.

4.4.3.3 Early Stopping and Backpropagation

Another approach to determine the parameters θ of a regularizer is by means of truncated optimization [54] also known as early stopping. This approach is motivated by bilevel optimization but instead of solving the lower-level problem exactly, only a certain number of iteration steps of a first-order method is performed to approximate the solution of the lower-level problem. Thus, the objective of the bilevel problem (4.10) changes to

$$\begin{aligned} &\min_\theta \mathbb{E}_{(y,\xi)\sim\mathcal{T}} \ell(x_S(y,\xi) - y) \\ &\text{s.t} \quad x_{s+1}(y,\xi) = x_s(y,\xi) - \tau \left(D_1 \mathrm{R}(x_s(y,\xi),\theta) + A^*(Ax_s(y,\xi) - z)\right) \end{aligned}$$

for $s = 0, \ldots, S-1$ and a given initial image estimate x_0. Here, $\tau > 0$ is a sufficiently small step size. The particular advantage of this discriminative learning approach is that the gradient of the loss function w.r.t. the regularizer's parameters can be directly computed by differentiation through the iterative scheme. Moreover, early stopping in conjunction with backpropagation enables an efficient inference and

training process. Recently, Mehmood and Ochs [199] showed that the gradient w.r.t. the parameters of the truncated optimization problem converges to the gradient w.r.t. the parameters of the bilevel problem under suitable assumptions. We provide further details on this learning strategy in Chapter 6.

4.5 Conclusion

To conclude this chapter, let us apply the parametric FoE and TDV regularizers to the Gaussian denoising problem of the water castle image. Figure 4.19 depicts the denoised water castle image using the FoE regularizer trained by truncated optimization and Figure 4.20 the corresponding result of the TDV regularizer. Compared to the results of TV (Figure 4.11), TGV (Figure 4.12), and the Student-t-based first-order regularizer (Figure 4.15), the image quality drastically improved. Both regularizers are able to denoise the homogeneous sky region, while preserving sharp edges. The TDV regularizer yields a smoother sky region and favors continuous edge lines, resulting in a realistically restored image as can be seen at the rooftop of the castle. In addition, the PSNR score increased by more than 1dB for the FoE regularizer and another 0.5dB for the TDV regularizer. Hence, image regularizers incorporating higher-order image features yield excellent restoration results.

Figure 4.19: Denoised water castle image using the FoE regularizer using 48 filters of size 7×7 and radial basis functions incorporating $N_w = 31$ weights. The parameters are determined by early stopping and backpropagation.

Figure 4.20: Denoised water castle image using the TDV_3^3 regularizer. The parameters are determined by early stopping and backpropagation.

5 Variational Networks

In this chapter, we discuss variational networks (VNs) that were introduced to study the connections between variational methods and today's successful deep learning approaches. The VNs are based on the solid theoretical foundations of incremental proximal methods, which enable the exploration of theoretical properties such as the limitations of convexity in the context of learning image regularizers for image restoration. We empirically demonstrate that the solutions of convex variational models for Gaussian image denoising and non-blind image deblurring are not capable of describing all facets of natural images and are outperformed by non-convex models. However, the results can be improved for convex and non-convex models by using parametrized incremental proximal methods.

In addition, we present an approach on how to extend VNs to reconstruct low-dose 3-dimensional helical computed tomography (CT) scans. For this task, we consider two dose reduction methods: X-ray tube current reduction and X-ray beam interruption also known as SparseCT. In the first case, we train a VN to denoise a current-reduced reconstruction to account for the smaller signal-to-noise ratio, whereas in the second case the VN learns a reconstruction scheme that suppresses undersampling artifacts. The numerical results indicate that the proposed VNs improve performance over state-of-the-art iterative model-based denoising and sparse reconstruction techniques. Moreover, VNs for SparseCT compare favorably to VNs for current reduction, particularly for reconstruction of small low-contrast features.

This chapter is based on the publications:

Erich Kobler, Teresa Klatzer, Kerstin Hammernik, and Thomas Pock. "Variational Networks: Connecting Variational Methods and Deep Learning". In: *German Conference on Pattern Recognition*. 2017

Erich Kobler, Matthew J. Muckley, Baiyu Chen, Florain Knoll, Kerstin Hammernik, Thomas Pock, Daniel K. Sodickson, and Ricardo Otazo. "Variational network learning for low-dose CT". In: *Proceedings of the 5th CT Meeting*. 2018

5.1 Connecting Variational Methods and Deep Learning

There has been a long tradition of using variational methods to tackle computer vision problems including denoising [1], deblurring [2, 201], segmentation [3, 202], tracking [203, 204] and optical flow [4] due to their simplicity, performance and profound theoretical foundations. In recent years, these approaches have been outperformed by deep learning methods. Despite the success of deep learning in computer vision [150, 205], it is unclear whether there exists a theoretical connection between variational methods and deep learning. In this paper, we try to answer this question by establishing relations between both worlds.

Variational methods are based on minimizing an energy function. A famous convex variational model (VM) for image restoration is the Rudin-Osher-Fatemi (ROF) model [1]. In the discrete setting it is defined as

$$\min_{x \in \mathbb{R}^n} \left\{ \mathrm{E}(x) \coloneqq \|Dx\|_{2,1} + \tfrac{\nu}{2} \|x - x_0\|_2^2 \right\}, \tag{5.1}$$

where $x \in \mathbb{R}^n$ represents an image with n pixels, $x_0 \in \mathbb{R}^n$ the noisy observation and $D \in \mathbb{R}^{2n \times n}$ is the linear operator that computes the discrete horizontal and vertical derivatives defined in Section 4.1.2. The solution set of (5.1) is characterized by the first-order optimality condition, i.e. $\partial \mathrm{E}(x) \ni 0$ or $\nabla \mathrm{E}(x) = 0$ if $\mathrm{E}(x)$ is continuously differentiable. Figure 5.1 visualizes the 2×2 patch statistics of this set along with the associated statistics of noisy and clean images, as described in Section 4.2.2. The solution set of (5.1) shows a significant difference to the true image statistics especially towards the polar regions, which suggests that the solution set of (5.1) cannot capture the diverse statistics of natural images.

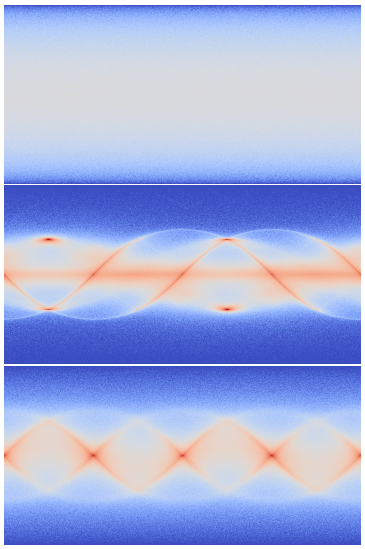


Figure 5.1: Estimated log–probability density of 2×2 image patches of the campus dataset [162] on the unit sphere in the zero-mean and contrast-normalized patch space for noisy patches (top), patches denoised by the ROF model (5.1) (middle), and ground truth patches (bottom). Note that all 3 densities are scaled identically.

A natural idea for improving the ROF model is to increase its flexibility by introducing additional terms. Chambolle and Lions [27] increased the model complexity by formulating image reconstruction as a convex infimal convolution problem. Another convex VM is the total generalized variation (TGV) [29], which extends the ROF model by *modeling* higher order statistics. However, Black and Anandan [186] demonstrated that incorporating non-convex functions improves results because the applied non-convex functions suppress outliers as known from robust statistics. They optimize the non-convex VMs using the graduated non-convexity method [41], which solves a sequence of VMs starting with a convex model that gradually becomes non-convex.

The idea of *learning* higher-order statistics to enhance the results of variational methods for image reconstruction was introduced by Roth and Black [51]. They proposed to learn a prior (regularization) consisting of an ensemble of filters together with corresponding non-convex potential functions called fields of experts (FoE) using contrastive divergence. Later, Kunisch and Pock [198] formulated the learning of regularization parameters of a VM as a bi-level optimization problem, which was extended in [53] to learn analysis operators of (non-)convex VMs including the FoE model. Their results on image denoising indicate that non-convex models perform best, confirming the findings of Zhu and Mumford [45]. Also, Domke [54] enhanced the performance of the FoE model by discriminatively learning incomplete energy minimization schemes that consist just of a few iterations inspired by [64]. The combination of

- unrolling a gradient descent scheme for the FoE model and
- abandoning energy minimization by parameterizing each step individually

led to the optimized nonlinear reaction-diffusion processes of Chen et al. [55], which improved the state-of-the-art on many reconstruction tasks [120, 206, 207].

In contrast, the deep learning community pursues a completely different approach to increase the model complexity to account for the diverse statistics of natural images. Since the early convolutional neural networks [144, 208], advances in network training and the use of more complex, deeper networks have led to remarkable results in many areas of computer vision, including classification [143, 150], and restoration [142, 209]. Increasing the model complexity by stacking more and more layers works just to some extent due to a degradation problem reported by He et al. [150]. To avoid this problem, they introduced residual networks that have a simple computational structure which eases the training of very deep models.

In this section, we introduce variational networks that originate from minimizing a parametric energy utilizing proximal incremental methods [81]. The VNs have the same computational structure as residual networks and thus are easy to train. Moreover, the concept of VNs enables us to explore theoretical properties such as the role of convexity in the field of natural image restoration. Therefore, we extend the

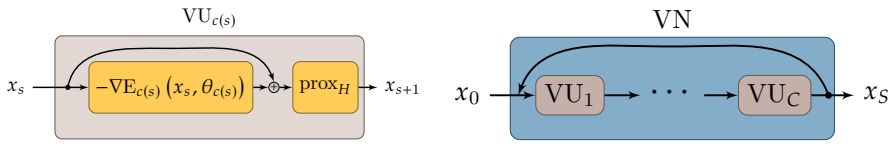


Figure 5.2: Illustration of our proposed variational units (5.3) (left) and their combination to a variational network (right) for a cyclic scheme.

FoE regularization structure by fully parametrized potential functions that can be trained either convex or non-convex.

5.1.1 Method

We propose VNs that are motivated by proximal gradient and proximal incremental methods and yield the same computation structure as residual networks. The basic structure of VNs evolves naturally by performing incremental proximal gradient steps [81] to solve problems of the form

$$\min_{x \in \mathbb{R}^n} \left\{ \mathrm{E}(x) \coloneqq \sum_{c=1}^{C} \mathrm{E}_c(x, \theta_c) + \mathrm{H}(x) \right\}, \tag{5.2}$$

where C defines the number of components, $x \in \mathbb{R}^n$ represents some data, i.e. an image, $\mathrm{E}_c : \mathbb{R}^n \to \mathbb{R}$ are smooth component functions parametrized by θ_c and $\mathrm{H} : \mathbb{R}^n \to \mathbb{R}$ is a convex, lower semi-continuous (l.s.c.) function. An incremental proximal gradient step is defined as

$$x_{s+1} = \mathrm{prox}_{\tau_s \mathrm{H}}\left(x_s - \tau_s \nabla \mathrm{E}_{c(s)}(x_s, \theta_{c(s)})\right), \tag{5.3}$$

where τ_s is the step size of the s-th step. We fix the component selection function $c(s) = \mathrm{mod}(s, C) + 1$ to obtain a cyclic procedure as depicted in Figure 5.2. We call the scheme (5.3) variational unit (VU) in analogy to residual units. The VU is the basic building block of a VN. The output of the C^{th} unit $x_{s=C}$ ends the first cycle. It is also the output of a corresponding residual network [150]. Moreover, VNs generalize the optimized nonlinear reaction-diffusion processes [55] as they can be interpreted as a single cycle of a parametrized incremental scheme.

5.1.1.1 Relation to Incremental Methods

The formulation of VNs is based on incremental proximal methods, which were proposed by Nedić and Bertsekas [81, 109]. These methods were designed to solve large-scale energy minimization problems consisting of smooth and non-smooth components. Such problems can be cast into the form

$$\min_{x \in \mathcal{X}} \left\{ \mathrm{E}(x) \coloneqq \mathrm{F}(x) + \mathrm{H}(x) = \sum_{c=1}^{C} \mathrm{E}_c(x) + \mathrm{H}(x) \right\}, \tag{5.4}$$

where $\mathcal{X} \subset \mathbb{R}^n$ is a convex set, F is the sum of the smooth components $\mathrm{E}_c : \mathbb{R}^n \to \mathbb{R}$ and $\mathrm{H} : \mathbb{R}^n \to \mathbb{R}$ accounts for the convex, l.s.c. and non-smooth parts. If the indicator function of $\mathcal{X}$ is included in $H(x)$, (5.4) becomes an unconstrained optimization problem. In analogy to [81], an incremental proximal gradient step is defined as

$$x_{s+1} = \mathrm{prox}_{\tau_s \mathrm{H}}\left(x_s - \tau_s \nabla \mathrm{E}_{c(s)}(x_s)\right), \tag{5.5}$$

where $\nabla E_{c(s)}(x_s)$ is the gradient of a single component selected by the function $c : \mathbb{N} \to \{1, \ldots, C\}$ and the proximal map is defined by

$$\mathrm{prox}_{\tau \mathrm{H}}(\widehat{x}) \coloneqq \operatorname*{arg\,min}_{x \in \mathbb{R}^n} \mathrm{H}(x) + \frac{1}{2\tau} \|x - \widehat{x}\|_2^2 .$$

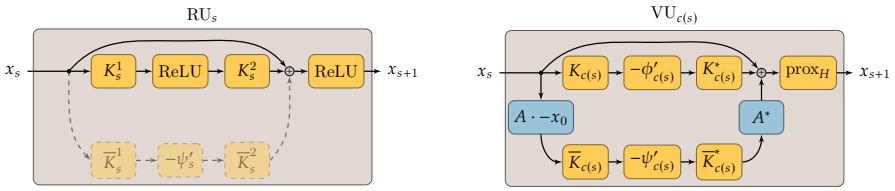


Figure 5.3: Visualization of the structural correspondence between (multi) residual units [212] (left) and variational units for image restoration (5.12) (right). Note the data term gradient of the variational unit can be interpreted as a second residual mapping in the data domain. The multi-residual unit is turned into a residual unit [150] by omitting the dashed path.

If F only consists of a single component, i.e. $\mathrm{F}(x) = \mathrm{E}_1(x)$, the scheme (5.5) simplifies to the proximal gradient method defined as

$$x_{s+1} = \operatorname{prox}_{\tau_s \mathrm{H}}\left(x_s - \tau_s \nabla \mathrm{E}_1(x_s)\right). \tag{5.6}$$

First, we assume that all components E_c are *convex*. In this case, Bertsekas [81] showed that the incremental proximal method (5.5) converges to a stationary point in the limit for a diminishing step size, satisfying $\sum_{s=0}^{\infty} \tau_s = \infty$, $\sum_{s=0}^{\infty} \tau_s^2 < \infty$, for both cyclic and random component selection $c(s)$. Moreover, he proved approximate convergence for a constant step size ($\tau_s = \tau > 0$). The assumptions of the proofs are fulfilled if all components E_c are Lipschitz continuous on $\mathcal{X}$.

If the components E_c are *non-convex*, one can still show approximate convergence of (5.5) in the limit using the inexact non-convex proximal splitting algorithm of Sra [107]. The requirements of Sra, i.e. all E_c have a Lipschitz continuous gradient on $\mathcal{X}$, imply that the component functions E_c are Lipschitz continuous on $\mathcal{X}$ since $\mathcal{X}$ is compact. Then (5.5) approximately converges to a stationary point for a constant step size $\tau_s = \tau > 0$.

5.1.1.2 Relation to Residual Networks

Deep residual networks were proposed by [150] to alleviate a degradation problem arising in deep neural network training, indicated by increasing training *and* test error despite growing model complexity. Residual networks circumvent this problem by stacking many simple residual units, which are characterized by

$$x_{s+1} = p(x_s + g_s(x_s)), \tag{5.7}$$

where $x_s \in \mathbb{R}^n$ is the input and $x_{s+1} \in \mathbb{R}^n$ the output of the s^{th} layer, $p : \mathbb{R}^n \to \mathbb{R}^n$ is a point-wise scalar function (*e.g.* rectified linear unit (ReLU)) and $g_s : \mathbb{R}^n \to \mathbb{R}^n$ are residual functions. Typically, these residual functions are defined as

$$g_s(x_s) = \sum_{i=1}^{N_r} K_{s,i}^2 a(K_{s,i}^1 x_s), \tag{5.8}$$

where the matrices $K_{s,i}^1, K_{s,i}^2 \in \mathbb{R}^{n \times n}$ model convolutions and N_r defines the number of convolution kernels. The function $a : \mathbb{R}^n \to \mathbb{R}^n$ is often the ReLU activation. The resulting networks can be efficiently trained for more than 1000 layers. The combination of the individual residual units forms a powerful ensemble of networks [210], yielding state-of-the-art results on challenging competitions, *e.g.* ImageNet [143] and MS COCO [104].

By comparing the structure of variational units (5.3) and residual units (5.7), we see that the proximal map in (5.3) corresponds to $p(x) = \mathrm{ReLU}(x)$ in (5.7) if H is the indicator function of the positive orthant. If we assume $\tau_s = 1$, then $g_s(x_s)$ corresponds to $-\nabla f_{c(s)}(x_s)$. This is either true for $s \leq C$ or if a residual network shares parameters in a periodic fashion [211]. To emphasize this structural resemblance, Figure 5.3 visualizes a residual and a variational unit. The residual function (5.8) corresponds

to a gradient if $K_{s,i}^2 = K_{s,i}^{1\top}$. If this relation is approximate ($K_{s,i}^2 \approxeq K_{s,i}^{1\top}$), g_s can still be interpreted as a gradient with error. Consequently, this type of network fits into the VN formulation and both networks have the same computational structure. Hence, VNs combine the practical benefits of residual networks, i.e. avoid the degradation problem, and the rich theory of incremental methods, including convergence and convex optimization theory.

5.1.2 Variational Networks for Image Restoration

We formulate image restoration as a variational energy minimization problem with a fully trainable regularization as well as data term and cast this problem into the VN formulation.

5.1.2.1 Problem Formulation and Parametrization

A variational model for image restoration in the form of (5.2) is given by

$$\min_{x \in \mathcal{X}} \left\{ \mathrm{E}(x) \coloneqq \sum_{c=1}^{C} \left\{ \mathrm{E}_c(x, \theta_c) \coloneqq \mathrm{R}_c(x, \theta_c) + \mathrm{D}_c(x, \theta_c) \right\} \right\}, \tag{5.9}$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$ represents an image constrained on $\mathcal{X} = \{x \in \mathbb{R}^n : 0 \leq x_i \leq b, \text{for } i = 1, \ldots, n\}$ with $b > 0$. The vector θ_c represents all parameters of each component. The regularization term $\mathrm{R}_c(x, \theta_c)$ models prior knowledge, whereas the data term $\mathrm{D}_c(x, \theta_c)$ models the data fidelity. The specific form of the FoE regularization term variant is given by

$$\mathrm{R}_c(x, \theta_c) = \sum_{i=1}^{N_r} \sum_{j=1}^{n} \phi_i^c \left((K_i^c x)_j \right), \tag{5.10}$$

where $\phi_i^c(x) : \mathcal{Y} \to \mathbb{R}$ are potential functions defined on $\mathcal{Y} = \{y \in \mathbb{R} : |y| \leq m\}$, their associated matrices $K_i^c \in \mathbb{R}^{n \times n}$ model convolutions of the image x with kernels k_i^c and N_r defines the number of regularization functions. Some learned kernel-function pairs are depicted in Figure 5.4. The convolution of an $s_k \times s_k$ kernel k_i^c can also be expressed as a matrix-vector multiplication Xk_i^c with the matrix $X \in \mathbb{R}^{n \times s_k^2}$ and the vector $k_i^c \in \mathbb{R}^{s_k^2}$.

We also parametrize the data term with kernel-function pairs to incorporate higher-order statistics in the data domain, as motivated by [213]. It is defined as

$$\mathrm{D}_c(x, \theta_c) = \sum_{i=1}^{N_d} \sum_{j=1}^{n} \psi_i^c \left(\left(\overline{K}_i^c (Ax - x_0) \right)_j \right), \tag{5.11}$$

where $x_0 \in \mathcal{X}$ describes the degraded observation and $A \in \mathbb{R}^{n \times n}$ models a linear operator. As before, the matrices $\overline{K}_i^c \in \mathbb{R}^{n \times n}$ model convolutions with kernels $\overline{k}_i^c$, $\psi_i^c(y) : \mathcal{Y} \to \mathbb{R}$ are the corresponding potential functions and N_d specifies the number of kernel-function pairs.

We define the VUs for image restoration akin to (5.3) as

$$x_{s+1} = \mathrm{proj}_{\mathcal{X}}(x_s - \tau_s \nabla \mathrm{E}_{c(s)}(x_s, \theta_{c(s)})), \tag{5.12}$$

where the proximal operator of (5.3) simplifies to the projection onto $\mathcal{X}$. The gradient of a selected component $\mathrm{E}_c(x, \theta_c)$ is given by

$$\nabla \mathrm{E}_c(x_s, \theta_c) = \sum_{i=1}^{N_r} K_i^{c\top} \phi_i'^c \left(K_i^c x_s \right) + A^* \sum_{i=1}^{N_d} \overline{K}_i^{c\top} \psi_i'^c \left(\overline{K}_i^c (Ax_s - x_0) \right). \tag{5.13}$$

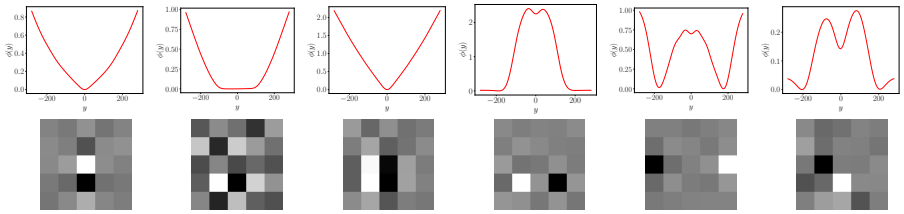


Figure 5.4: Sample kernel-function pairs $(k_i^c, \phi_i^c(y))$ of the trained VNs. The left three pairs are convex samples, whereas the right three were extracted from non-convex VNs.

Since we learn the influence functions $\phi_i'^c(y)$ and $\psi_i'^c(y)$, we can fix the step size $\tau_s = 1$, as it is reflected in the scale of both influence functions. Due to the above parametrization, all the component functions E_c of the corresponding VN are smooth, Lipschitz continuous functions over $\mathcal{X}$ with bounded and Lipschitz continuous gradient over $\mathcal{X}$ as long as the functions $\phi_i'^c(y)$ and $\psi_i'^c(y)$ fulfill these constraints. The proofs can be found in [56]. Note that the runtime and memory requirements of the VNs resemble those of [55], since the basic operations are identical.

5.1.2.2 Training

To train the VNs for image restoration we parametrize the influence functions $\phi_i'^c(y)$ and $\psi_i'^c(y)$ in analogy to [55, 192] with radial basis functions

$$\phi_i'^c(y) = \sum_{j=1}^{N_w} \exp\left(-\frac{(y-\mu_j)^2}{2\sigma^2}\right) w_{ij}^c, \tag{5.14}$$

where w_{ij}^c are the individual basis weights that correspond to a single radial basis (μ_j, σ) and N_w defines the number of basis functions. To shorten notation we group the coefficients into $w_i^c = (w_{i1}^c, \dots, w_{iN_w}^c)^\top$. The functions $\psi_i'^c(x)$ are parametrized in the same way by $\overline{w}_i^c$. We group the parameters of a single component c into the vector $\theta_c = (k_1^c, w_1^c, \dots, k_{N_r}^c, w_{N_r}^c, \overline{k}_1^c, \overline{w}_1^c, \dots, \overline{k}_{N_d}^c, \overline{w}_{N_d}^c)$. The parameters of all components are gathered into $\theta = (\theta_c, c = 1 \dots C)$. We define the training cost functional for N input-target pairs (x_0^i, y^i) as

$$\min_{\theta \in \Theta} \left\{ \mathrm{J}(\theta) \coloneqq \frac{1}{N} \sum_{i=1}^{N} \left\| x_S^i(\theta) - y^i \right\|_1 \right\}, \tag{5.15}$$

where x_S^i is the output after S steps (5.12). We use the ℓ^1-norm because of its robustness [214]. In addition, we constrain the parameters θ to be in an admissible set Θ. This set ensures that the kernels k_i^c and $\overline{k}_i^c$ have zero mean and ℓ^2-norm one to avoid a scaling problem as outlined in [55]. Θ also allows us to incorporate constraints on the functions $\phi_i^c(y)$ and $\psi_i^c(x)$ such as convexity by defining suitable conditions for w_i^c and $\overline{w}_i^c$ as shown below. Note that if all $\phi_i^c(y)$ and $\psi_i^c(x)$ are convex, the entire energy (5.9) becomes convex [90]. We optimize the non-convex training problem (5.15) with the inertial incremental proximal method (IIPG) defined in Algorithm 6, where $\delta_\Theta(\theta)$ is the indicator function of the admissible set Θ.

For image restoration, we introduce the following constraints on the parameters. We enforce that the convolution kernels k_i^c and $\overline{k}_i^c$ have zero mean and are normalized, i.e.

$$k_i^c, \overline{k}_i^c \in \mathcal{K} = \left\{ k \in \mathbb{R}^{s_k^2} : 1^\top k = 0, \|k\|_2 = 1 \right\}, \tag{5.16}$$

in order to ensure that the domain $\mathcal{Y}^n$ of the convolution result $(K_i^c x)$ is bounded and symmetric around zero. The proximal map for the kernels in Algorithm 6 simplifies to the projection on $\mathcal{K}$ which can be simply computed by subtracting the mean followed by normalization.

Algorithm 6: Inertial incremental proximal gradient (IIPG) algorithm.

Input: Training set $\mathcal{S}$, step size α, number of epochs N_E and number of mini batches N_B

Partition $\mathcal{S}$ into N_B minibatches $\mathcal{S} = \bigcup_{b=1}^{N_B} \mathcal{B}_b$
Choose initial parameters θ^0
$\theta^1 = \theta^0$
$l = 1$
for $e = 1$ **to** N_E **do**
  for $b = 1$ **to** N_B **do**
    Perform over-relaxation
    $\tilde{\theta} = \theta^l + \frac{l-1}{l+2}(\theta^l - \theta^{l-1})$
    Compute gradient on $\mathcal{B}_b$
    $g^l = \frac{\partial L(\tilde{\theta})}{\partial \theta}$
    Compute preconditioning P^l by (5.17) *and* (5.18)
    Perform proximal gradient descent step
    $\theta^{l+1} = \operatorname{prox}^{P^l}_{\alpha\delta(\Theta)}(\tilde{\theta} - \alpha P^l g^l)$
    $l = l + 1$

To speed up Algorithm 6, we use a diagonal block-wise preconditioning matrix P^l given by

$$P^l = \operatorname{diag}\left(P^l_{k_1^1}, P^l_{w_1^1}, \ldots, P^l_{k^C_{N_k}}, P^l_{w^C_{N_k}}, P^l_{\lambda^C}\right), \tag{5.17}$$

where the diagonal matrices P^l_p for the individual parameters are defined by

$$P^l_p = \left\| \frac{\partial L(\theta)}{\partial p} \right\|_2^{-1} \mathrm{Id}, \tag{5.18}$$

where $p \in \{k_i^c, w_i^c, \lambda^c\}$ and Id is the corresponding identity matrix.

Our goal is to investigate the limitations of convexity due to its property that each local minimum is a global minimum. To this end, we need to learn convex potential functions $\rho_i^c(y)$. Their domain is a compact subset of $\mathcal{Y} \subset \mathbb{R}$ because the input image elements are bounded ($x \in \{x \in \mathbb{R} : 0 \leq x \leq b\}$) and the kernels have norm one. Thus, $\mathcal{Y} = \{y \in \mathbb{R} : |y| \leq b\}$ is a convex set. Since the potential functions are scalar, a sufficient condition for convexity is

$$\phi_i^{c\prime\prime}(y) \geq 0 \quad \forall y \in \mathcal{Y}. \tag{5.19}$$

Hence, we need to ensure that $\phi_i^{c\prime\prime}$ is positive over $\mathcal{Y}$. Its is given by

$$\phi_i^{c\prime\prime}(y) = -\sum_{j=1}^{N_w} \frac{(y-\mu_j)}{\sigma^2} \exp\left(-\frac{(y-\mu_j)^2}{2\sigma^2}\right) w_{ij}^c,$$

which can be shortened in matrix-vector notation to

$$\phi_i^{c\prime\prime}(y) = \Phi_i^{c\prime\prime}(y) w_i^c,$$

where the matrix $\Phi_i^{c\prime\prime}(y) \in \mathbb{R}^{n \times N_w}$ holds coefficients for each radial basis. Since we cannot test the convexity condition (5.19) for all elements in $\mathcal{Y}$, we define control points $y_p \in \mathcal{Y}^{N_p}$. In practice it turned out that $N_p = 2N_w + 1$ yields enough control points to ensure convexity of $\phi_i^c(y)$ on $\mathcal{Y}$ due to the overlap of the individual radial basis functions. Consequently, the weights w_i^c of an influence function $\phi_i^c(y)$ have to lie in the set

$$w_i^c \in \mathcal{W} = \left\{ w \in \mathbb{R}^{N_w} : -\Phi^{c\prime\prime}(y_p) w \leq 0 \right\}.$$

Type	Corresponding scheme
$\text{VN}_N^{1,t}$	proximal gradient method (5.6) (energy minimization)
$\text{VN}_N^{C,t}$	proximal incremental method (5.5) (energy minimization)
$\text{VN}_N^{t,t}$	single cycle proximal incremental method (5.5) (residual network)

Table 5.1: Overview of the VN types. The subscript N defines the number of used kernel-function pairs $N_r = N$. The superscript specifies the number of components C and the step t for which the VN was optimized.

We can easily incorporate this constraint in the proximal map of Algorithm 6 for w_i^c:

$$w_i^{c,l} = \text{prox}_{\tau\delta(\mathcal{W})}(z) = \underset{-\Phi^{c\prime\prime}(y_p)w \leq 0}{\arg\min} \frac{1}{2} \|w - z\|_2^2 \tag{5.20}$$

with $z = w_i^{c,l-1} - \alpha P_{w_1^1}^l \frac{\partial L}{\partial w_i^c}$. We add Lagrange multipliers $p \in \mathbb{R}^{N_c}$ to transform (5.20) into the saddle point problem

$$\min_{w \in \mathbb{R}^{N_w}} \max_{p \geq 0} \frac{1}{2} \|w - z\|_2^2 - p^\top \Phi^{c\prime\prime}(y_p) w. \tag{5.21}$$

Its closed form solution w.r.t. w is

$$w = z + \Phi^{c\prime\prime}(y_p)^\top p. \tag{5.22}$$

By plugging this into (5.21) and rearranging terms, we get the quadratic problem

$$\min_{p \in \mathbb{R}^{N_c}} \frac{1}{2} \left\| -\Phi^{c\prime\prime}(y_p)^\top p - z \right\|_2^2 \quad \text{s.t.} \quad p \geq 0, \tag{5.23}$$

which can be efficiently solved by FISTA [99]. The proximal gradient step of w_i (5.22) can be performed with the minimizer of (5.23). Note that the quadratic problem (5.23) must be solved in every iteration of Algorithm 6. However, the problem can be easily parallelized for all potential functions, which helps to keep the overhead for convex functions minimal.

5.1.3 Experiments

We conduct three groups of experiments to show the versatility of VNs and to explore the role of convexity. Table 5.1 defines all used VN types and outlines their relation to the previously discussed methods. We conduct all experiments for denoising and non-blind deblurring. In the case of denoising, the degraded input x_0 is a noisy observation and the linear operator A in (5.11) simplifies to an identity operation. For non-blind deblurring, the input is a blurry and noisy observation, and the linear operator A models a convolution with a known blur kernel. The denoising VNs (N-VN) use just a single data term $N_d = 1$ and an identity kernel $\bar{k}_1^{-1}$, while the deblurring VNs (B-VN) apply $N_d = N_r$ kernel-function pairs. To train VNs for both problems, we use 400 training patches of size 180×180 extracted from the BSDS500 train and test sets [215]. We generate the noisy training inputs by adding white Gaussian noise with $\sigma = 25$ to the clean images. To generate the blurry training data, we extract 11×11 motion blur kernels from [216], convolve them with the clean training patches, and add 1% white Gaussian noise. The test sets are generated in the same way for denoising and non-blind deblurring. We use 68 images from the BSDS500 [215] validation set and the motion blur kernels from [2] to ensure that neither the images nor the blur kernels are used during training.

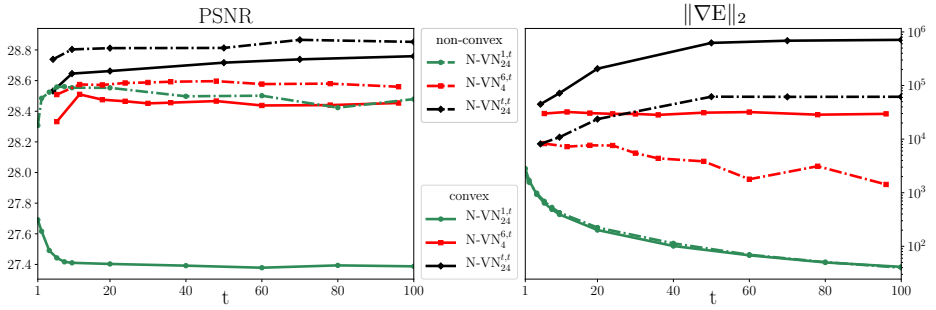


Figure 5.5: Average PSNR curves on the test set of the trained VN types for Gaussian image denoising along with the gradient norm of the corresponding energy $E(x_s)$.

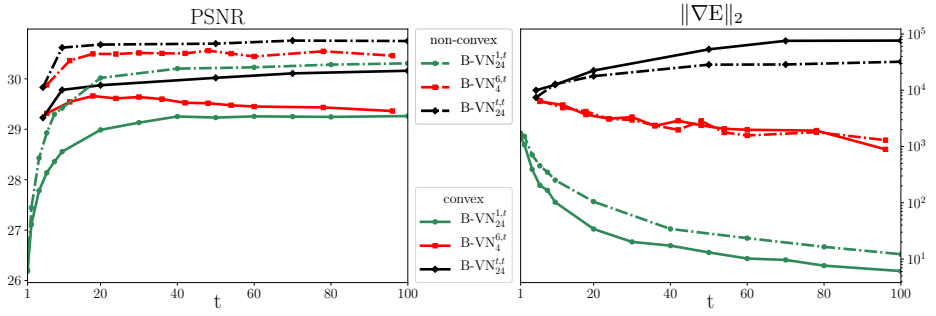


Figure 5.6: Average PSNR values and corresponding gradient norm on the test set of the different VN types for non-blind deblurring.

5.1.3.1 Energy Minimization with Variational Networks

In the first experiment, we set up VNs to perform energy minimization following the proximal gradient method (5.6) by fixing the number of components to $C = 1$, i.e. $E(x) = E_1(x)$. For both denoising and non-blind deblurring, we train convex and non-convex VNs up to $t = 100$ steps. The resulting PSNR values and the ℓ^2-norm of the gradients $\|\nabla E(x_s)\|_2$ are depicted in green color in Figure 5.5 and 5.6. As expected, the decreasing gradient norm with increasing steps t indicates that the methods actually minimize an underlying energy.

The PSNR curves for denoising (Figure 5.5) differ for convex and non-convex N-VN$_{24}^{1,s}$. The performance of the non-convex VNs increases initially and slowly declines with increasing s, while the convex N-VN$_{24}^{1,s}$ yields the best results after a single step. This indicates that a convex regularization of the form (5.10) is not a good prior for natural images because by approaching a minimizer (increasing t) the results become worse. Surprisingly, the highly parametrized convex N-VN$_{24}^{1,s}$ performs marginally better than the ROF model for $s > 10$. Note that all convex schemes are local optima of the non-convex training problem (5.15). In the case of non-blind deblurring the PSNR curves (Figure 5.6) are similar for convex and non-convex B-VN$_{24}^{1,s}$. Both VNs require more steps to yield satisfactory results since deblurring is a harder problem than denoising. Nevertheless, the non-convex B-VN$_{24}^{1,s}$ outperform the convex ones by a large margin (1dB).

5.1.3.2 Incremental Minimization with Variational Networks

In a second experiment, we evaluate the performance of VNs that follow an incremental energy minimization scheme as in (5.5). We use $C = 6$ components and $N_r = 4$ kernel-function pairs. Thus, the number of parameters of a cycle is approximately the same as in the previous experiment. The resulting PSNR values as well as the gradient norm for the trained convex and non-convex VN$_4^{6,s}$ are depicted in red color in Figure 5.5 for denoising and in Figure 5.6 for non-blind deblurring.

In contrast to the previous experiment, the PSNR curves for denoising and deblurring are rather flat for both convex and non-convex VN$_4^{6,s}$. Thus, they manage to generate good results after just a few steps and maintain the quality with increasing t. However,

	ROF [1]	convex			non-convex			BM3D [73]	$\text{TNRD}^5_{5\times5}$ [55]
		$\text{VN}^{1,t}_{24}$	$\text{VN}^{6,s}_{4}$	$\text{VN}^{s,s}_{24}$	$\text{VN}^{1,t}_{24}$	$\text{VN}^{6,s}_{4}$	$\text{VN}^{s,s}_{24}$		
denoising	27.39	27.69	28.51	*28.76*	28.56	28.60	**28.87**	28.56	*28.78*
non-blind deblurring	28.35	29.26	29.66	30.16	30.31	*30.56*	**30.76**	-	-

Table 5.2: Average PSNR values on the test set for the VN types. The reported PSNR values are computed using the best performing depth t of each VN type.

the results after 100 steps are far from approaching a stationary point, as indicated by the rather slowly decreasing gradient norm $\|\nabla \text{E}\|_2$. This effect is very strong for the convex N-$\text{VN}^{6,s}_4$ because these VNs learn a sequence of components that alternate between strong blurring and detail recovery from the data term, leading to large gradients. In terms of PSNR values, this behavior yields superior results compared to the first experiment. The decreasing PSNR value of the convex B-$\text{VN}^{6,s}_4$ with increasing depth may originate from local optima of the learning problem.

5.1.3.3 Variational Networks in a Residual Setup

In the final experiment, we investigate the performance of VNs in a residual network or trainable nonlinear reaction-diffusion setting [55], i.e. each step (5.12) has its own parameter set θ_s ($C = s$). Hence, the number of parameters increases linearly with the depth of the $\text{VN}^{s,s}_{24}$. These VN types can still be interpreted as incremental proximal methods that apply each component just once.

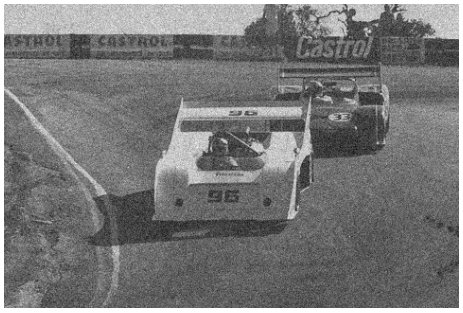
The increasing model complexity with growing t leads to a steady increase of the performance for the $\text{VN}^{s,s}_{24}$ on both restoration tasks, depicted in Figure 5.5 and 5.6. The gradient norm also grows along with the depth t due to the additional components. Consequently, these VNs do not minimize a corresponding energy. However, they yield the best performance on the image restoration tasks as shown in Table 5.2. In contrast to Chen et al. [55], our findings on image denoising suggest that the shape of the learned potential functions (Figure 5.4) is of little importance since the convex and non-convex N-$\text{VN}^{s,s}_{24}$ perform almost equally well, as shown in Table 5.2. The convex N-VNs rather require the flexibility of incremental schemes in order to yield satisfactory results. Still, convexity seems to be a limiting factor for non-blind deblurring since all convex VNs perform worse than the non-convex ones.

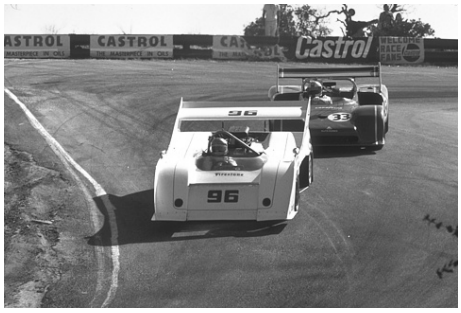
To also qualitatively assess the performance of the proposed VNs, we depict the resulting reconstruction for image denoising in Figure 5.7 and for deblurring in Figure 5.8. In general, the non-convex models yield visually more pleasing results. Especially, the convex VN trained as a truncated energy minimization suffers from large residual noise artifacts.

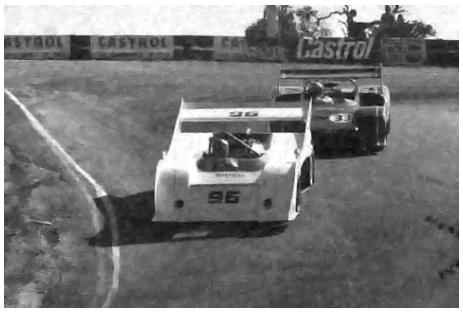
5.1.4 Conclusion

In this section, we established a link between variational energy minimization methods and deep learning approaches by introducing variational networks. The VNs consist of stacked parametrized incremental proximal steps that have the same favorable computational structure as residual units. We demonstrated that the versatile VN formulation can be used to learn proximal gradient schemes, incremental proximal schemes as well as residual networks and optimized reaction-diffusion processes. Moreover, our parametrization of the VNs for image restoration allows us to learn corresponding *convex* energies.

We used this novel possibility to evaluate the limitations of convexity in the context of natural image restoration. Our findings on denoising and non-blind deblurring show that our convex formulations yield inferior results compared to non-convex formulations. Additionally, the incremental VN types require just a few steps (layers) to yield reasonable results even for the challenging task of non-blind deblurring.


noisy


target


convex $\text{VN}_{24}^{1,6}$


non-convex $\text{VN}_{24}^{1,6}$


convex $\text{VN}_{24}^{6,12}$


non-convex $\text{VN}_{24}^{6,48}$


convex $\text{VN}_{24}^{100,00}$


non-convex $\text{VN}_{24}^{100,100}$

Figure 5.7: Qualitative results of the various VN types for image denoising. Note that the convex VNs generate artifacts in smooth regions, whereas the non-convex VNs avoid those.

noisy

target

convex $\text{VN}_{24}^{1,100}$

non-convex $\text{VN}_{24}^{1,100}$

convex $\text{VN}_{24}^{6,18}$

non-convex $\text{VN}_{24}^{6,18}$

convex $\text{VN}_{24}^{100,00}$

non-convex $\text{VN}_{24}^{100,100}$

Figure 5.8: Qualitative results of the various VN types for non-blind image deblurring. Note that the results of the convex VNs seem to be a bit more noisy than the non-convex results.

5.2 Application to Low-dose Computed Tomography Reconstruction

The increasing utilization of CT scanners in clinical imaging examinations has triggered the need to reduce the radiation dose, particularly for recurrent studies. One of the most common approaches is to reduce the tube current, e.g., tube current modulation [217], or lower tube currents in conjunction with iterative model-based denoising methods [218]. These techniques have been successfully integrated in commercial scanners, but they only offer moderate radiation dose reductions of 30-40% in practice due to compromises between denoising and smoothing.

The radiation dose can also be mitigated without reducing the tube current by decreasing the number of X-rays that penetrate a patient during a CT scan. The compressed sensing (CS) theory [18] supports this approach, since CT images are compressible in a transform domain and reducing the number of X-ray projections results in small additive incoherent streaking artifacts. A simple way to omit projections is to perform angular undersampling, i.e., just acquire projections for a fraction of the angular views, as proposed by Chen et al. in [219]. The SparseCT method [83] extended this idea by blocking a subset of X-rays in an incoherent way across the angular and slice dimensions, which divides the overall undersampling along multiple dimensions and thus increases the performance of CS for the reconstruction of the whole volume.

Recent low-dose CT reconstruction algorithms for low-current and/or undersampled data are typically model-based iterative methods that incorporate prior knowledge to increase image quality. These prior models are typically rather simple and model just a small subset of the CT image statistics, e.g., the popular total variation (TV) prior enforces sparsity in the image gradient domain. In addition, the balance between a regularizing prior term and a data fidelity term has to be empirically tuned to generate suitable reconstructions. In accelerated magnetic resonance imaging, deep learning was introduced to overcome this empirical tuning and to learn image models that are tailored towards medical imaging, demonstrating significant improvements over standard compressed sensing algorithms [120]. Likewise, recent work on deep learning for low-dose CT demonstrated improved performance compared to standard denoising and sparse reconstructions [193, 194, 220]. The U-net-like structures advocated in [193, 220] as well as the residual encoding network introduced in [194] learn a mapping from low-dose filtered back-projection images to reference images that encodes and decodes the relevant information in contrast to the step-wise refinement structure of [120].

In this work, we propose to learn variational networks for low-dose CT data acquired with tube current reduction and SparseCT. We train the VNs on four clinical abdominal datasets and evaluate the reconstruction quality of the proposed VNs on a test dataset and compare it to state-of-the-art model-based reconstructions.

5.2.1 Model-based CT Reconstruction

The process of acquiring CT data of a volume $x \in \mathbb{R}^n$ can be formalized as

$$z = Ax + \xi, \tag{5.24}$$

where $z \in \mathbb{R}^l$ is the post-log measured data of l X-ray projections. Here, $\xi \in \mathbb{R}^l$ models the effects of quantum and electronic noise and is assumed to be Gaussian due to preprocessing. The linear forward operator $A : \mathbb{R}^n \to \mathbb{R}^l$ implements the mapping from the volume to the measurement data that is defined by the scanner geometry. For SparseCT A additionally implements the undersampling pattern.

For a given noisy and possibly undersampled CT scan data z, the inverse problem of recovering the volume x is usually defined by a variational minimization problem such as

$$\min_{x \in \mathbb{R}^n} \left\{ \mathrm{E}(x) \coloneqq \nu \left\| \nabla x \right\|_1 + \frac{1}{2} \left\| Ax - z \right\|_2^2 \right\}. \tag{5.25}$$

Here the scalar $\nu \geq 0$ is used to balance the solution between smoothness, which is enforced by the anisotropic TV, i.e., ℓ^1-norm of the image gradients, and data fidelity. A suitable algorithm to solve (5.25) is the primal-dual approach with line search [102], since it requires just a few evaluations of the operator A that are computationally expensive.

5.2.2 Variational Networks for CT

Gradient-based optimization schemes of variational models in imaging, such as (5.25), can often be viewed as recurrent neural networks (RNNs). This observation, inspired Chen and Pock [55] to train all parameters of a gradient descent scheme for variational image reconstruction models, i. e. analysis operators, potential functions, weighting, and step sizes, from data. VNs [56] connect this scheme, convolutional neural networks and variational minimization. To adapt VNs for CT, we apply FoE-type regularizers [51] of the form

$$\mathrm{R}_c(x, \theta_c) = \sum_{i=1}^{N_r} \sum_{j=1}^{n} \left(\phi_i^c((K_i^c x)_j) \right) \tag{5.26}$$

that are parameterized by 3-dimensional convolution operators $K_i^c : \mathbb{R}^n \to \mathbb{R}^n$ and corresponding potential functions $\phi_i^c : \mathbb{R} \to \mathbb{R}$ for $i = 1, \ldots, N_r$. These functions are point-wisely applied to the associated filter response and are parameterized by weighted radial basis functions facilitating the learnable weights $w_j^c \in \mathbb{R}^{N_w}$, as defined in (5.14).

We use this prior model to construct a variational energy that fits into the VN framework [56] and define it as

$$\mathrm{E}_{\{\mathrm{TCR,SCT}\}} \coloneqq \sum_{c=1}^{C} \mathrm{E}^c_{\{\mathrm{TCR,SCT}\}}(x), \tag{5.27}$$

$$\mathrm{E}^c_{\{\mathrm{TCR,SCT}\}}(x) = \mathrm{R}_c(x) + \frac{\lambda_c}{2} \mathrm{D}_{\{\mathrm{TCR,SCT}\}}(x), \tag{5.28}$$

where the data term $D_{\{\mathrm{TCR,SCT}\}}(x)$ is adapted according to the dose-reduction approach. In the case of tube current reduction (TCR) we learn to denoise initial low-dose reconstruction, hence we use a simple ℓ^2-norm denoising data term

$$\mathrm{D}_{TCR}(x) = \left\| x - x_0 \right\|_2^2. \tag{5.29}$$

In the case of sparse computed tomography (SCT) we use the forward operator A and the undersampled data z to enforce data consistency to the undersampled data and facilitate the reconstruction scheme

$$\mathrm{D}_{SCT}(x) = \left\| Ax - z \right\|_2^2. \tag{5.30}$$

For both low-dose VNs we use a cyclic component selection function, i.e., $c(s) = \mathrm{mod}\,(s, C)$, and follow [56] to define a variational unit (VU) as

$$x_s = x_{s-1} - \nabla \mathrm{E}^{c(s)}_{\{TCR,SCT\}}(x_{s-1}), \tag{5.31}$$

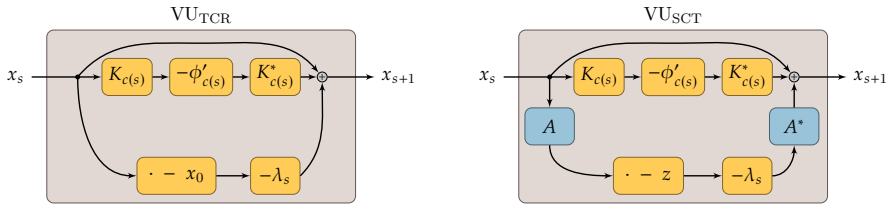


Figure 5.9: Illustration of the variational units (VU) for CT denoising (left) and CT reconstruction (right).

where the gradients of the energy components are given by

$$\nabla \mathrm{E}^c_{TCR}(x) = \sum_{i=1}^{N_r} (K^c_i)^* \phi^{c\prime}_i(K^c_i x) + \lambda_c (x - x_0),$$

$$\nabla \mathrm{E}^c_{SCT}(x) = \sum_{i=1}^{N_r} (K^c_i)^* \phi^{c\prime}_i(K^c_i x) + \lambda_c A^*(Ax - z).$$

The adjoint operator of K^c_i is denoted as $(K^c_i)^*$ and it is defined as a convolution with all 180° rotated filter kernels followed by a point-wise summation. Likewise, A^* denotes the adjoint operator of the forward operator A. Figure 5.9 illustrates the computational structure of the building blocks of a VN for low-dose CT. The input x_0 is transformed into the output x_S by applying S steps of the form (5.31).

5.2.2.1 Training of VNs for CT

To train a VN for a set of training samples $(x^i_0, y^i)_{i=1}^N$, we minimize the problem

$$\min_{\theta \in \Theta} \frac{1}{2} \sum_{i=1}^{N} \left\| b^s \odot (x^i_S - y^i) \right\|_2^2, \tag{5.32}$$

where $\theta = (w^c_i, K^c_i, \lambda_c,\ c = 1, \ldots, C, i = 1, \ldots, N_r)$ encompasses all parameters of the VN. Following [56], we constrain the parameters to an admissible set Θ that enforces $\lambda_c \geq 0$ and that each convolution filter has zero mean and it's ℓ^2-norm lies on the unit ball. We are only interested in reconstructing the central scan regions because of the missing ray density at border regions. Thus, we apply a binary mask $b^s \in \{0,1\}^n$ that selects the 9 central slices where $y \in [0,1]$ and $\odot$ indicates a point-wise multiplication. Note that we rescaled the images such that the Hounsfield units (HU) interval $[-200, 280]$ is mapped to $[0,1]$ to ease training and account for the desired HU range. We solve the constrained training problem (5.32) by using the Adam optimizer [110] extended by an additional backprojection step onto Θ after each gradient step. We perform 1000 gradient steps using the default moments of the Adam optimizer and a step size of 10^{-2}.

5.2.2.2 Experimental Setup

For the reconstruction of low-dose CT data we apply $S = C = 10$ variational units and use $N_r = 32$ convolution filters of size $11 \times 11 \times 3$. Their corresponding activation functions are parameterized by $N_w = 31$ Gaussian radial basis functions. We scaled the volumes for both training and test data such that the interesting HU interval $[-200, 280]$ is mapped onto $[0,1]$ to ease the training of the parameters. We use 8 filter-function-pairs that are defined on the interval $[-4, 4]$ to regularize the entire HU range, whereas the remaining 24 filter-function-pairs are defined on $[-1, 1]$ to account for the details in the desired tissue interval. In total 126,090 parameters were trained for each VN.

We used four clinical 3-dimensional in-vivo abdominal CT scans of different patients of a Siemens Definition AS scanner. Table 5.3 shows the acquisition properties of the train and test scans. In order to fit the CT data reconstruction onto a single GPU, we

	reference tube current mAs	tube voltage kV	radiation dose CTDIvol	gantry rotations -
train	240	120	21.19	16
	240	120	19.01	26
	240	120	22.26	19
	350	120	29.63	20
test	320	100	12.90	17

Table 5.3: Training and test datasets for CT.

SAFIRE [218]	TV	VN TCR	VN SCT
17.75 ± 2.11	8.84 ± 1.20	7.91 ± 0.90	7.72 ± 0.82

Table 5.4: Quantitative comparison of the different 1/4-dose CT methods by means of RMSE to the target y, measured in HU.

split the data of each CT scan after a full gantry rotation and ended up with 81 batches for training and 17 test samples. For every sample, we reconstructed an imaged volume of size $384 \times 384 \times 30$. The target volumes y^i were computed by solving (5.25) with $\nu = 1$ using the primal dual algorithm with linesearch [102] on the full-dose CT data. Likewise, the initial reconstructions x_0^i were generated with $\nu = 10^{-9}$ using either simulated fully-sampled low-dose data [221] or binary subsampled full-dose data for SCT. We apply the same W1S4 undersampling pattern as in [222] for a 4-fold dose reduction.

5.2.3 Numerical Results

We used the test dataset to evaluate the reconstruction quality of the learned VNs for both TCR and SCT for 4-fold radiation dose reduction. Table 5.4 depicts a quantitative evaluation of the root mean squared error (RMSE) of the proposed VNs and state-of-the-art model-based denoising and reconstruction approaches. In Figure 5.10, we qualitatively compare representative abdominal slices reconstructed by the proposed VNs to the full-dose reference, SAFIRE [218] and TV reconstruction.

In the case of tube current reduction, the proposed VN for TCR outperforms SAFIRE [218] in terms of RMSE and also in reconstruction quality. The VN presents a higher noise reduction of the imaged volume, while keeping the fine structures of the vessels in the liver. The resulting images are slightly smoothed though. Since SAFIRE applies an edge-enhancing kernel to highlight edges in the reconstructions, we removed the skin region from the binary mask b in the evaluation process to perform a fair comparison. Figure 5.11 depicts the difference to a corresponding reference slice for the considered methods. Clearly, SAFIRE yields higher differences at edge regions, but also the remaining regions are rather noisy.

In the case of SparseCT, the trained VN yields a lower RMSE than the TV model-based reconstruction using 4-fold undersampled test data. The VN for SCT removes the aliasing artifacts better than the TV reconstruction, while maintaining the fine vessels in the liver. Moreover, the reconstructions of the VN for SCT present more details than those of the VN for TCR and are also sharper, highlighting the advantages of SparseCT over tube current reduction for the same dose reduction factor. In addition, the reconstructions of a VN for SCT using 6-fold undersampling are shown on the right in Figure 5.10. Despite the increased dose reduction, the VN for SCT is able to reconstruct the fine details and remove aliasing artifacts and yield reconstructions with similar quality.

5.2.4 Conclusion

In this section, we extended variational networks to reconstruct CT volumes from low-dose data. We learned VNs for two popular radiation dose reduction methods, namely tube current reduction and SparseCT. The proposed VNs yield reconstructions that

outperform state-of-the-art denoising and sparse reconstruction methods for low-dose CT. The VNs present a higher noise and artifact reduction while fine details such as vessels are properly reconstructed. The learned reconstructions for undersampled data (SparseCT) show more details and are sharper than the learned denoising scheme for reduced-current data. Our experiments suggest that the proposed VNs increase the image quality for a given radiation dose and would enable higher radiation dose reductions. Future work includes the extension of the binary undersampling masks of SparseCT to more realistic undersampling masks as in [223].

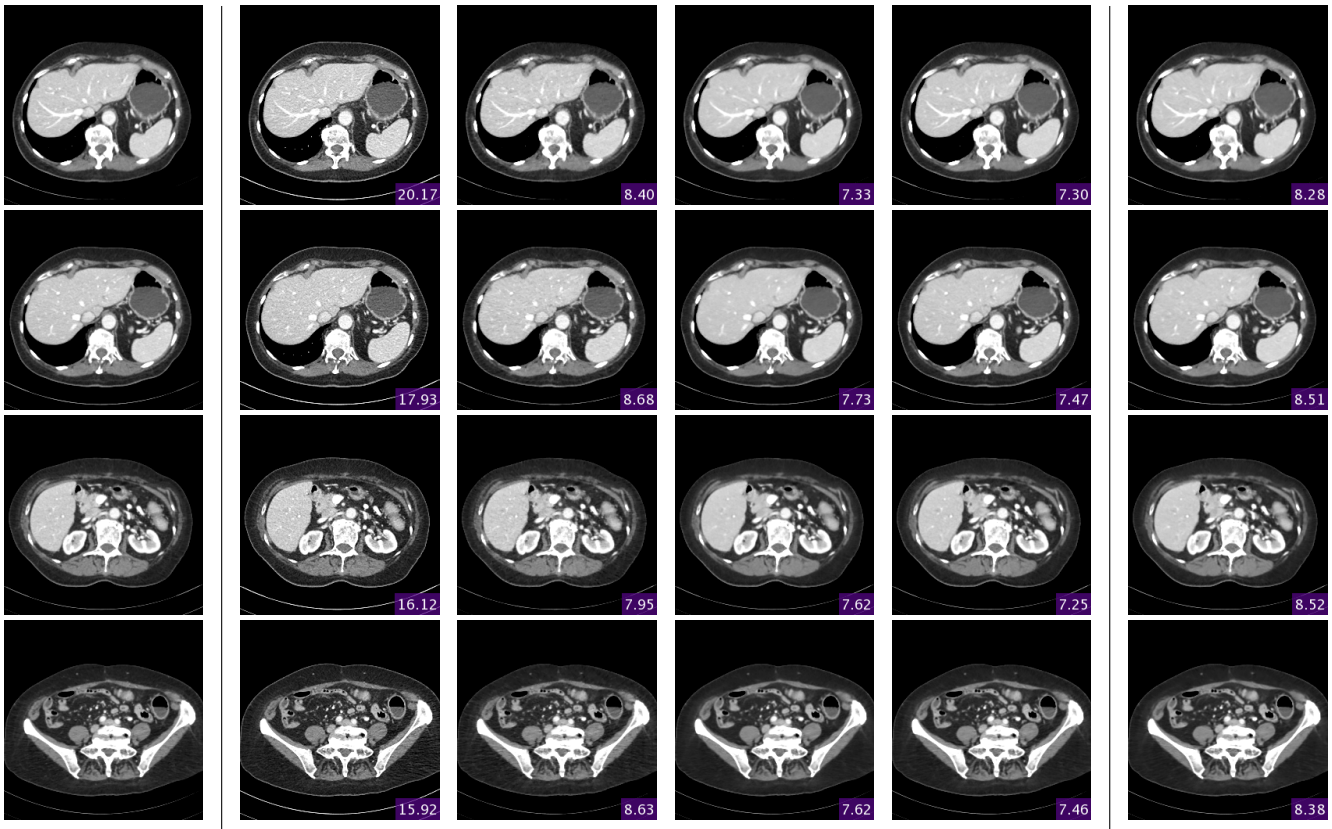


Figure 5.10: Representative slices for reconstruction of in vivo abdominal test data for low-dose CT. The purple boxes report RMSE values. (a) Target: TV ($\nu = 1$) reconstruction of the fully-sampled high dose data, (b) SAFIRE [218] using 1/4 dose, (c) TV ($\nu = 1.75$) reconstruction using 4-fold undersampling, (d) VN for TCR reconstruction using $S = 10$ steps and 1/4 dose, (e) VN for SCT reconstruction using $S = 10$ steps and 4-fold undersampling, and (f) VN for SCT reconstruction using $S = 10$ steps and 6-fold undersampling.

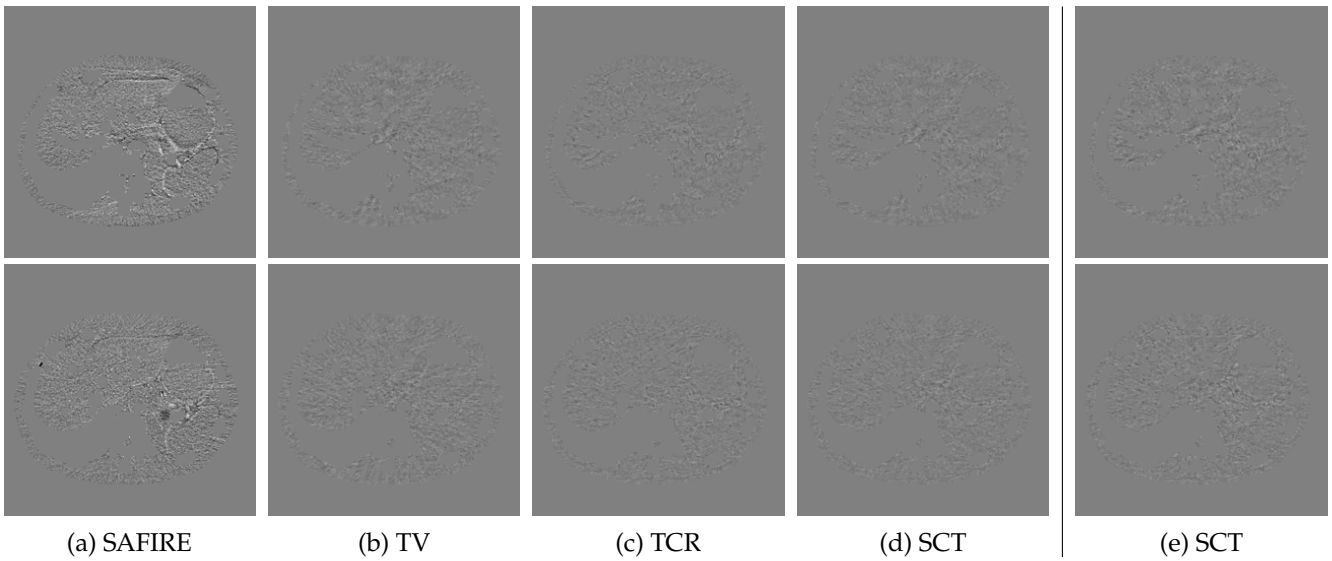


Figure 5.11: Error to the reference reconstruction y for the first two slices presented in Figure 5.10. (a) SAFIRE [218] using 1/4 dose, (b) TV ($\nu = 1.75$) reconstruction using 4-fold undersampling, (c) VN for TCR reconstruction using $S = 10$ steps and 1/4 dose, (d) VN for SCT reconstruction using $S = 10$ steps and 4-fold undersampling, and (e) VN for SCT reconstruction using $S = 10$ steps and 6-fold undersampling. Note that we mapped the HU interval $[-150, 150]$ to $[0, 1]$ to ease visualization of the error.

Mean-field Optimal Control Parameter Estimation

6

In this chapter, we investigate a well-known phenomenon of variational approaches in image processing, where typically the best image quality is achieved when the gradient flow process is stopped before converging to a stationary point. This paradox originates from a tradeoff between optimization and modeling errors of the underlying variational model and holds true even if deep learning methods are used to learn highly expressive regularizers from data. We take advantage of this paradox and introduce an optimal stopping time into the gradient flow process, which in turn is learned from data along with the regularizer's parameters by means of a mean-field optimal control approach, where a gradient flow on the underlying variational energy defines the state equation. After a time-discretization of the gradient flow, we obtain variational networks (VNs), which can be interpreted as a particular type of recurrent neural networks (RNNs). We prove the existence of solutions of both the time-continuous and time-discrete mean-field optimal control problem for a class of regularizers, including the fields of experts (FoE) and total deep variation (TDV) regularizer. We also derive a first-order condition to verify the optimality of the stopping time. Moreover, we perform a stability analysis with respect to the initial values and the parameters of the regularizers and experimentally verify the robustness against adversarial attacks and numerically derive upper bounds for the generalization error. Finally, using the general-purpose TDV regularizer we achieve state-of-the-art results for numerous imaging tasks such as image denoising, single image super-resolution (SISR), accelerated magnetic resonance imaging (MRI) reconstruction, and undersampled computed tomography (CT) reconstruction.

This chapter is based on the publications:

Alexander Effland, Erich Kobler, Karl Kunisch, and Thomas Pock. "Variational Networks: An Optimal Control Approach to Early Stopping Variational Methods for Image Restoration". In: *J. Math. Imaging Vision* 62.3 (2020)

Erich Kobler, Alexander Effland, Karl Kunisch, and Thomas Pock. "Total Deep Variation for Linear Inverse Problems". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2020

Erich Kobler, Alexander Effland, Karl Kunisch, and Thomas Pock. "Total Deep Variation: A Stable Regularizer for Inverse Problems". In: *IEEE Trans. Pattern Anal. Mach. Intell.* (2020). (submitted)

Erich Kobler, Baiyu Chen, Alexander Effland, Thomas Pock, Daniel K. Sodickson, and Ricardo Otazo. "Total Deep Variation for SparseCT Reconstruction". In: *Proceedings of the 6th CT Meeting*. 2020

As in the previous chapters, we assume that the observations z are generated by a linear inverse problem of the form

$$z = Ay + \xi,$$

where y is the unknown ground truth, A is a known task-dependent linear operator and ξ is additive noise. For example, A is the identity matrix in the case of denoising, and it is a downsampling operator in the case of single image super-resolution. In

analogy to Chapter 4, we aim at computing the maximum a-posteriori (MAP) estimate by minimizing the energy

$$\mathrm{E}(x, z) := \mathrm{D}(x, z) + \mathrm{R}(x),$$

which is composed of the data fidelity term D and the regularizer R. While the data fidelity term is frequently straightforward to model, there has been much effort to design a regularizer that captures the complexity of the statistics of natural images.

A classical and widely used regularizer is the total variation (TV) first utilized in imaging by [1]. The TV is based on the first principle assumption that images are piecewise constant with sparse gradients. A well-known caveat of the sparsity assumption of TV is the formation of clearly visible artifacts known as staircasing effect. To overcome this problem, the first principle assumption has later been extended to piecewise smooth images incorporating higher-order image derivatives such as infimal convolution-based models [27] or the total generalized variation [29]. Inspired by the fact that edge continuity plays a fundamental role in the human visual perception, regularizers penalizing the curvature of level lines have been proposed in [225–227]. While these regularizers are mathematically well-understood, the complexity of natural images is only partially reflected in their formulation. For this reason, handcrafted variational methods have nowadays been largely outperformed by purely data-driven methods as predicted by Levin and Nadler [228] a decade ago.

It has been recognized quite early that a proper statistical modeling of regularizers should be based on learning [46], which has recently been advocated e.g. in [61, 63]. One of the most successful early approaches is the FoE regularizer [51], which can be interpreted as a generalization of the total variation, but builds upon learned filters and learned potential functions. While the FoE prior was originally learned generatively, it was shown in [52] that a discriminative learning via implicit differentiation yields improved performance. A computationally more feasible method for discriminative learning is based on unrolling a finite number of iterations of a gradient descent algorithm [54]. Additionally using iteration-dependent parameters in the regularizer was shown to significantly increase the performance (TNRD [55], [229]). In [56], VNs are proposed, which give an incremental proximal gradient interpretation of TNRD. Interestingly, such truncated schemes are not only computationally much more efficient but are also superior in performance with respect to the full minimization. A continuous-time formulation of this phenomenon was proposed in [57] by means of an optimal control problem, within which an optimal stopping time is learned.

The significance of stability for data-driven methods has recently been addressed in [230], in which a systematical treatment of adversarial attacks for inverse problems was performed. This issue has been studied in the context of classification by [231], where adversarial samples have been introduced. These samples are computed by perturbing input images as little as possible such that the attacked algorithm predicts wrong labels. Incorporating adversarial samples in the training process of data-driven methods increases their robustness as studied in [231]. In recent years, several methods were proposed for generating adversarial examples such as the fast gradient sign method [232], Deepfool [209], or universal adversarial perturbations [233]. In the context of inverse problems, adversarial examples are typically computed by maximizing the norm difference between the output of an algorithm and the associated ground truth in a local neighborhood around an input. Thus, the robustness w.r.t. adversarial attacks of an algorithm depends to a large extent on the local Lipschitz constant of the mapping defined by the algorithm.

In this chapter, we minimize an energy composed of a data fidelity term and a parametric regularizer using a gradient flow emanating from the corrupted input image on a finite time horizon $[0, T]$ for a stopping time T, where the terminal state of the gradient flow defines the reconstructed image. Then, the stopping time and

the parameters of the regularizer are computed by solving a mean-field optimal control problem as advocated in [58], in which the cost functional is defined as the expectation of the loss function with respect to a training data distribution. The state equation of the optimal control problem is a stochastic ordinary differential equation coinciding with the gradient flow of the energy, where the only source of randomness is the initial state. We prove the existence of minimizers for this optimal control problem using the direct method in the calculus of variations. A semi-implicit time-discretization of the gradient flow results in a discretized optimal control problem in the mean-field setting, for which we also prove the existence of minimizers as well as a first-order necessary condition to automatize the computation of the optimal stopping time. This training process is a form of *discriminative learning* because we directly learn the functional form of the negative log-posterior [234], in which the regularizer can be interpreted as a *discriminative prior*. In fact, the learned regularizers adapt to the specific imaging task as we show in the eigenfunction analysis. Moreover, the particular recursive structure of the discrete gradient flow allows the derivation of a stability analysis with respect to the initial states and the learned regularizer parameters. Both estimates depend on the local Lipschitz constant of the regularizer, which is estimated in the mean-field setting. Several numerical experiments demonstrate the applicability of the proposed method to numerous image restoration problems, in which we obtain state-of-the-art results using the TDV regularizer, defined in Section 4.4.2. In particular, we examine the robustness of this approach against perturbations and adversarial attacks, and an upper bound for the generalization error is empirically computed.

The major contributions are as follows:

- a rigorous mathematical analysis including a mean-field optimal control formulation of the learning problem,
- a stability analysis of the proposed method, which is validated by numerical experiments,
- a nonlinear eigenfunction analysis for the visualization and understanding of the learned regularizer,
- numerical evaluation of the robustness against adversarial attacks and empirical upper bounds for the generalization error,
- state-of-the-art results on a number of classical image restoration and medical image reconstruction problems with an impressively low number of learned parameters.

6.1 Early Stopping

The benefit of early stopping for iterative algorithms is examined in the literature from several perspectives. In the context of ill-posed inverse problems, early stopping of iterative algorithms is frequently considered and analyzed as a regularization technique. There is a variety of literature on the topic and we therefore only mention the selected monographs [25, 235–237]. Frequently, early stopping rules for inverse problems are discussed in the context of the Landweber iteration [238] or its continuous analog commonly referred to as Showalter's method [239] and are based on criteria such as the discrepancy or the balancing principle.

In what follows, we provide an overview of recent advances related to early stopping. Raskutti et al. [240] exploit early stopping for non-parametric regression problems in reproducing kernel Hilbert spaces (RKHS) to prevent overfitting and derive a data-dependent stopping rule. Yao et al. [241] discuss early stopping criteria for gradient descent algorithms for RKHS and relate these results to the Landweber iteration. Quantitative properties of the early stopping condition for the Landweber iteration are presented in Binder et al. [242]. Zhang and Yu [243] prove convergence and consistency results for early stopping in the context of boosting. Prechelt [244] introduces several

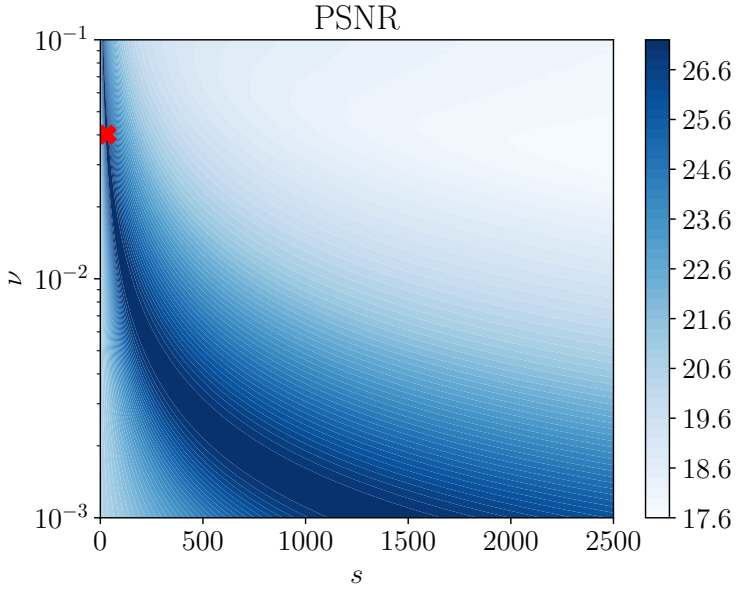


Figure 6.1: Contour plot of the PSNR score depending on the number of iteration steps s and the regularization parameter ν for ROF denoising. The global maximum $(31, 0.04)$ is marked with a red cross.

heuristic criteria for optimal early stopping based on the performance of the training and validation error. Rosasco and Villa [245] investigate early stopping in the context of incremental iterative regularization and prove sample bounds in a stochastic environment. Matet et al. [246] exploit an early stopping method to regularize (strongly) convex functionals. In contrast to these approaches, we propose early stopping on the basis of finding a local minimum with respect to the time horizon of a properly defined energy.

To illustrate the necessity of early stopping for iterative algorithms, we revisit the established ROF (TV-L^2) denoising functional [1], which amounts to minimizing the variational problem

$$\mathrm{E}(x) = \nu \left\|Du\right\|_1 + \tfrac{1}{2}\left\|x - z\right\|_2^2$$

among all images $x \in \mathbb{R}^n$ of size $n = n_1 \times n_2$, where D is the finite difference gradient operator defined in Section 4.1.2, $\nu > 0$ is the regularization parameter and $z \in \mathbb{R}^n$ refers to a corrupted input image. An elementary, yet very inefficient optimization algorithm relies on a gradient descent using a smoothed regularizer ($\beta > 0$)

$$\mathrm{E}_\beta(x) = \nu \sum_{i=1}^{n} \sqrt{|(D_h x)_i|^2 + |(D_v x)_i|^2 + \beta^2} + \tfrac{1}{2}\left\|x - z\right\|_2^2, \tag{6.1}$$

where D_h and D_v denote the horizontal and vertical gradient operators with Neumann boundary constraint, for details see Section 4.1.2. For a comprehensive list of state-of-the-art methods to efficiently solve TV-based variational problems, we refer the reader to [10]. Figure 6.1 depicts the dependency of the peak signal-to-noise ratio (PSNR) score on the number of iterations and the regularization parameter ν for the ROF problem (6.1) using a step size 10^{-4} and $\beta = 10^{-6}$, where the input image $z \in [0,1]^n$ with a resolution of 512×512 is corrupted by additive Gaussian noise with standard deviation $\sigma = 0.1$. As a result, for each regularization parameter ν there exists a unique optimal number of iterations where the signal-to-noise ratio peaks. Beyond this point, the quality of the resulting image is deteriorated by staircasing artifacts and fine texture patterns are smoothed out. The global maximum $(31, 0.04)$ is marked with a red cross. The associated image sequence for the considered image[1] is shown in Figure 6.2 at the top. In addition, a second image sequence for $\nu = 0.001$ is depicted at the bottom. Both sequences yield similar PSNR scores for x_{31} ($\nu = 0.04$) and x_{1500} ($\nu = 0.001$). However, the first sequence requires only 31 steps and is thus much more computationally efficient.

1: image by Nichollas Harrison (CC BY-SA 3.0)

If the gradient descent is considered as a discretization of a time-continuous evolution

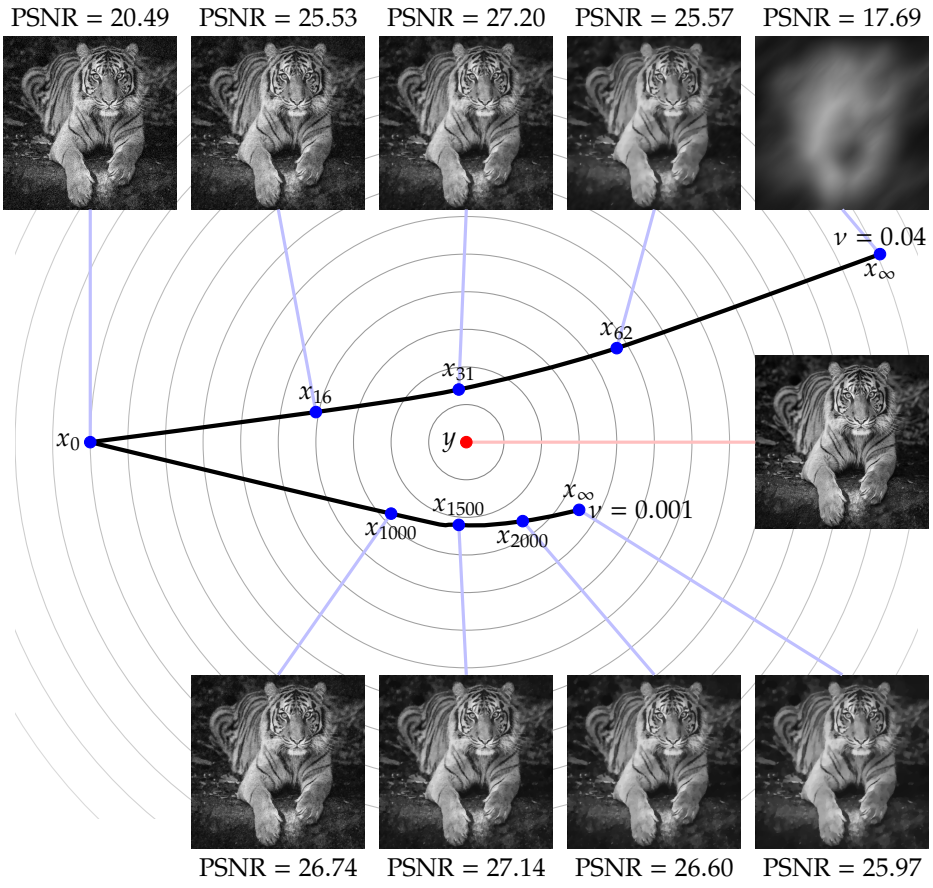


Figure 6.2: Illustration of early stopping for the ROF model (6.1) for $\nu = 0.04$ (top) and $\nu = 0.001$ (bottom). The blue dots mark selected steps which are annotated by the corresponding intermediate results x_s. The PSNR values are computed w.r.t. the ground truth y indicated by the red dot.

process governed by a differential equation, then the optimal number of iterations translates to an optimal stopping time.

6.2 Mean-field Optimal Control Approach to Early Stopping

In this section, we derive a time-continuous analog of static variational networks as gradient flows of an energy functional E composed of a data fidelity term D and a parametric regularizer R. The resulting ordinary differential equation is used as the state equation of a mean-field optimal control problem [58], in which the cost functional incorporates the squared L^2-distance of the state evaluated at the optimal stopping time to the ground truth and the control parameters are given by the learnable parameters of the regularizer.

Let $(\mathcal{Y} \times \Xi, \mathcal{F}, \mathbb{P})$ be a complete probability space on $\mathcal{Y} \times \Xi$ with σ-algebra $\mathcal{F}$ and probability measure $\mathbb{P}$. We denote by (y, ξ) a pair of independent random variables modeling the data representing the *ground truth image* $y \in \mathcal{Y} \subset \mathbb{R}^{nC}$ and *additive noise* $\xi \in \Xi \subset \mathbb{R}^l$ with associated distribution denoted by $\mathcal{T} = \mathcal{T}_\mathcal{Y} \times \mathcal{T}_\Xi$. Each ground truth image y represents an image of size $n = n_1 \times n_2$ with C channels and is related to the additive noise ξ by means of the *observation*

$$z = Ay + \xi,$$

where $A \in \mathbb{R}^{l \times nC}$ is a fixed task-dependent linear operator of this linear inverse problem. In particular, we assume that both $\mathcal{Y}$ and Ξ are compact sets, which implies that all observations are contained in a compact set $\mathcal{Z} \subset \mathbb{R}^l$.

To estimate the unknown ground truth image y from the observation z we pursue a

variational approach, which amounts to minimizing the energy functional

$$\mathrm{E}(x, z, \theta) \coloneqq \mathrm{R}(x, \theta) + \mathrm{D}(x, z) \tag{6.2}$$

among all $x \in \mathbb{R}^{nC}$. Here, we consider the squared ℓ^2-data fidelity term

$$\mathrm{D}(x, z) = \frac{1}{2} \|Ax - z\|_2^2$$

and for the parametric regularizer R we apply either the FoE [51] or TDV [59, 60] regularizer, which both depend on learned parameters $\theta \in \Theta$, where $\Theta \subset \mathbb{R}^p$ is compact and convex.

Let $\tilde{x} \in C^1([0, T], \mathbb{R}^{nC})$ be an *image trajectory*, which evolves according to the *gradient flow equation* [247] associated with (6.2) on a finite time interval $(0, T)$ given by

$$\dot{\tilde{x}}(t) = -D_1\mathrm{E}(\tilde{x}(t), z, \theta) = f(\tilde{x}(t), z, \theta) \coloneqq -D_1\mathrm{R}(\tilde{x}(t), \theta) - A^*(A\tilde{x}(t) - z) \tag{6.3}$$

for $t \in (0, T)$ and $x(0) = x_0$, where D_i is the gradient operator w.r.t. the i^{th} argument. Here, the observation-dependent initial value x_0 is computed as $x_0 = A_{\text{init}} z$ for a fixed task-dependent matrix $A_{\text{init}} \in \mathbb{R}^{nC \times l}$, which could be, for instance, the pseudoinverse of A. The proper choice of the stopping time $T \in [0, T_{\max}]$ for a fixed $T_{\max} > 0$ is essential for the quality of the reconstruction $\tilde{x}(T)$ of y. A more feasible, yet equivalent gradient flow is derived from the reparametrization $x(t) = \tilde{x}(tT)$, which yields for $t \in (0, 1)$

$$\dot{x}(t) = T f(x(t), z, \theta) \tag{6.4}$$

with the same initial value as before. We frequently write $x(t, y, \xi, T, \theta)$ to highlight the dependency of the image trajectory on the parameters $(y, \xi, T, \theta) \in \mathcal{Y} \times \Xi \times [0, T_{\max}] \times \Theta$ for given $t \in [0, 1]$. In particular, $x(1, y, \xi, T, \theta)$ is the computed output image. We remark that in contrast to [153, 156], inverse problems for image restoration rather than image classification are examined. Thus, we incorporate in (6.4) the classical gradient flow with respect to the full energy functional in order to promote data consistency, whereas in the classification tasks, only the gradient flow with respect to the regularizer is considered.

In what follows, the training process is described as a *mean-field optimal control problem* [58] with control parameters θ and T. To this end, let $\ell \in C^1(\mathbb{R}^{nC}, \mathbb{R}_0^+)$ be a convex and coercive *loss function*. Then, we define the *cost functional* as

$$\mathrm{J}(T, \theta) \coloneqq \mathbb{E}_{(y,\xi)\sim\mathcal{T}} \left[\ell(x(1, y, \xi, T, \theta) - y) \right],$$

which results in the mean-field optimal control problem

$$\inf_{T \in [0, T_{\max}], \theta \in \Theta} \mathrm{J}(T, \theta) \tag{6.5}$$

subject to the state equation (6.4). The particular choice of the cost functional originates from the observation that a visually appealing image restoration is obtained as the closest point on the trajectory of the gradient flow (reflected by the L^2-distance) to the ground truth y, as motivated for the ROF model in Figure 6.2. In this example, we seek the trajectory that is closest to the ground truth y in terms of the squared Euclidean distance among all trajectories of the ordinary differential equation (6.4) emanating from a constant initial value x_0. Note that in this case the entire trajectory is uniquely defined by the choice of the regularization weight ν. In contrast, in the mean-field optimal control problem (6.5) we seek the stopping time T and the parameters θ that result in the smallest Euclidean distance of the state at the stopping time $x(1)$ and the ground truth y on average over the data distribution $\mathcal{T}$. Also in this case, the trajectory emanating from an initial value x_0 is uniquely defined by the regularizer and its parameters θ.

Remark 6.2.1 The mean-field optimal control formulation already encompasses the *sampled optimal control problem*. In detail, given a finite training set $(y^i, \xi^i)_{i=1}^N \sim \mathcal{T}^N$ drawn from the data distribution we can define the *discrete probability measure* as $\mathbb{P}(y, \xi) = \frac{1}{N} \sum_{i=1}^N \delta[y = y^i]\delta[\xi = \xi^i]$, where $\delta[s = t] = 1$ if $s = t$ and 0 otherwise. This particular choice results in the sampled cost functional

$$\mathrm{J}(T, \theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(x(1, y^i, \xi^i, T, \theta) - y^i).$$

In this work, we consider parametric regularizers $\mathrm{R} : \mathbb{R}^{nC} \times \Theta \to \mathbb{R}$ that aggregate the local variation $\mathrm{r} : \mathbb{R}^{nC} \times \Theta \to \mathbb{R}^n$, i.e.

$$\mathrm{R}(x, \theta) = \sum_{i=1}^{n} \mathrm{r}(x, \theta)_i \tag{6.6}$$

with the particular structure

$$\mathrm{r}(x, \theta) = \Psi(Kx, \theta_\Psi),$$

where $\theta = (K, \theta_\Psi)$ denotes the entity of all parameters and

- $K \in \mathbb{R}^{nm \times nC}$ is the matrix representation of a learned convolution kernel for m feature channels with zero-mean constraint, i.e. $\sum_{j=1}^{nC} K_{i,j} = 0$ for $i = 1, \dots, nm$, which implies a spatial and radiometrical shift-invariance,
- $\Psi \in C^2(\mathbb{R}^{nm} \times \Theta_\Psi, \mathbb{R}^n)$ is a parametric twice continuously differentiable function with parameters in a compact set $\theta_\Psi \in \Theta_\Psi$, where we assume that $\|D_1\Psi\|_{C^0(\mathbb{R}^{nm})} \leq C_\Psi(\theta_\Psi)$.

The well-known FoE regularizer [51] aggregates the local energy of m convolution kernels $K_j \in \mathbb{R}^{n \times nC}$ and potential function $\phi \in C^2(\mathbb{R} \times \Theta_\phi, \mathbb{R})$ pairs by

$$\mathrm{R}_{\mathrm{FoE}}(x, \theta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \phi((K_j x)_i, w_j),$$

where each potential function is parameterized by radial basis functions with associated weights $w_j \in \Theta_\phi$, which lie in a compact set $\Theta_\phi \subset \mathbb{R}^{N_w}$. The FoE regularizer can be cast into the form

$$\mathrm{R}_{\mathrm{FoE}}(x, \theta) = \sum_{i=1}^{n} \Psi(Kx, \theta_\Psi)_i$$

as discussed in Section 4.4.1. The uniform boundedness assumption $\|D_1\Psi\|_{C^0(\mathbb{R}^{nm})} \leq C_\Psi(\theta_\Psi)$ is fulfilled if the radial basis functions ϕ have compact support. In the remainder, we use the shorthand notation FoE_m^k for an FoE regularizer using m convolution kernels each of size $k \times k$ and $N_w = 31$ radial basis functions.

The TDV regularizer extracts local features and combines them on multiple scales and in successive blocks in a convolutional neural network (CNN) and is defined by

$$R_{\mathrm{TDV}}(x, \theta) = \sum_{i=1}^{n} \overline{\Psi}(w\mathcal{N}(Kx))_i,$$

where $\mathcal{N} : \mathbb{R}^{nm} \to \mathbb{R}^{nm}$ is a multi-scale CNN, $w \in \mathbb{R}^{n \times nm}$ is a learned 1×1 convolution kernel, and $\overline{\Psi} : \mathbb{R}^n \to \mathbb{R}^n, (x_1, \dots, x_n) \mapsto (\psi(x_1), \dots, \psi(x_n))$ determines the energy using the *potential function* $\psi \in C^2(\mathbb{R}, \mathbb{R})$. We denote by TDV_a^b for integers $a, b \geq 1$ a TDV regularizer consisting of b blocks, where each of them has a U-Net [196] type architecture operating on a scales. For further details and a visualization of its computational structure, we refer to Section 4.4.2. The TDV regularizer also fits into

the considered class of regularizers (6.6) by simply choosing

$$\Psi(x, \theta_\Psi) = \overline{\Psi}(w\mathcal{N}(x)),$$

where θ_Ψ hold all the parameters of $\mathcal{N}$ and w. Then,

$$\begin{aligned}\|D_1\Psi(x, \theta_\Psi)\|_2 &= \left\|D\mathcal{N}(x)\, w\, D\overline{\Psi}(w\mathcal{N}(x))\right\|_2 \\ &\leq \|D\mathcal{N}(x)\|_2 \|w\|_2 \left\|D\overline{\Psi}(w\mathcal{N}(x))\right\|_2 \leq C_\Psi(\theta_\Psi),\end{aligned}$$

where we require that $\left\|D\overline{\Psi}\right\|_{C^0(\mathbb{R}^n)}$ is uniformly bounded, which is the case for all considered potential functions, and $\|D\mathcal{N}\|_{C^0(\mathbb{R}^{mn})}$ is uniformly bounded. In detail, the CNN $\mathcal{N}$ in the TDV regularizer is a complex concatenation of residual blocks, where the norm of the gradient of each of these blocks is given by

$$\left\|D_1\mathrm{R}^i_{j,k}(x, \theta)\right\|_2 = \left\|\mathrm{Id} + (K^i_{j,k,1})^* D\Phi(K^i_{j,k,1}x)K^i_{j,k,2}\right\|_2 \tag{6.7}$$

for $x \in \mathbb{R}^{nm}$. In particular, (6.7) can be uniformly bounded independently of x due to $\sup_{x\in\mathbb{R}} |\phi'(x)| = \frac{1}{2}$.

The existence of solutions to the mean-field optimal control problem for the particular structure of the regularizer (6.6) is established in the next theorem.

Theorem 6.2.1 *The minimum in* (6.5) *is attained.*

Proof. The particular structure of the considered parametric regularizers results in the estimate

$$\begin{aligned}\|D_1\mathrm{R}(x, \theta)\|_2 &= \|K^* D_1\Psi(Kx, \theta_\Psi)\|_2 \\ &\leq \|K\|_2 \|D_1\Psi(Kx, \theta_\Psi)\|_2 \leq \|K\|_2 C_\Psi(\theta_\Psi) =: C_\mathrm{R}(\theta)\end{aligned}$$

for all $x \in \mathbb{R}^{nC}$ and all $\theta \in \Theta$, where we used that $\|D_1\Psi\|_{C^0(\mathbb{R}^{nm})} \leq C_\Psi(\theta_\Psi)$. Then, the right-hand side of the state equation can be bounded as follows:

$$\left\|Tf(x, z, \theta)\right\|_2 \leq T(\|A\|_2 \|z\|_2 + C_\mathrm{R}(\theta) + \|A\|_2^2 \|x\|_2) \tag{6.8}$$

for $z \in \mathbb{R}^l$. This affine growth already ensures that the maximum domain of existence of the state equation (6.4) coincides with $\mathbb{R}$ [88, Theorem 2.17]. As a further result, we obtain that $x \in C^1([0, 1], C^0(\mathcal{Y} \times \Xi \times [0, T_{\max}] \times \Theta, \mathbb{R}^{nC}))$ due to the smoothness of the regularizer and

$$x(t, y, \xi, T, \theta) \in \mathcal{X}$$

for all $(t, y, \xi, T, \theta) \in [0, 1] \times \mathcal{Y} \times \Xi \times [0, T_{\max}] \times \Theta$ for a compact and convex set $\mathcal{X} \subset \mathbb{R}^{nC}$.

Let $(T^j, \theta^j) \in [0, T_{\max}] \times \Theta$ be a minimizing sequence for J with an associated state $x^j := x(\cdot, \cdot, \cdot, T^j, \theta^j) \in C^1([0, 1], C^0(\mathcal{Y} \times \Xi, \mathbb{R}^{nC}))$. The compactness of $[0, T_{\max}] \times \Theta$ implies that there exists a subsequence (not relabeled) such that $(T^j, \theta^j) \to (T^*, \theta^*) \in [0, T_{\max}] \times \Theta$. In what follows, we prove that x^j converges to $x^* := x(\cdot, \cdot, \cdot, T^*, \theta^*) \in C^1([0, 1], C^0(\mathcal{Y} \times \Xi, \mathbb{R}^{nC}))$ in $C^0([0, 1] \times \mathcal{Y} \times \Xi)$. We denote by L_x and L_θ the Lipschitz constants of $D_1\mathrm{R}$ w.r.t. x and θ, i.e.

$$\begin{aligned}\|D_1\mathrm{R}(x, \theta) - D_1\mathrm{R}(\widetilde{x}, \theta)\|_2 &\leq L_x \|x - \widetilde{x}\|_2, \\ \left\|D_1\mathrm{R}(x, \theta) - D_1\mathrm{R}(x, \widetilde{\theta})\right\|_2 &\leq L_\theta \left\|\theta - \widetilde{\theta}\right\|_2\end{aligned}$$

for all $x, \widetilde{x} \in \mathcal{X}$ and all $\theta, \widetilde{\theta} \in \Theta$. Then, we can estimate for any $(y, \xi) \in \mathcal{Y} \times \Xi$ and

$z = Ay + \xi$ as follows:

$$\begin{aligned}
&\left\|T^* f(x^*(t,y,\xi),z,\theta^*) - T^j f(x^j(t,y,\xi),z,\theta^j)\right\|_2 \\
\leq &|T^* - T^j| \left(\|A\|_2^2 \max_{x\in\mathfrak{X}} \|x\|_2 + \|A\|_2 \max_{z\in\mathfrak{Z}} \|z\|_2 + \max_{(x,\theta)\in\mathfrak{X}\times\Theta} \|D_1 \mathrm{R}(x,\theta)\|_2 \right) \\
&+ T_{\max} L_\theta \left\|\theta^* - \theta^j\right\|_2 + T_{\max}(\|A\|_2^2 + L_x) \left\|x^*(t,y,\xi) - x^j(t,y,\xi)\right\|_2 \\
=: &C_T |T^* - T^j| + C_\theta \left\|\theta^* - \theta^j\right\|_2 + C_x \left\|x^*(t,y,\xi) - x^j(t,y,\xi)\right\|_2 .
\end{aligned}$$

Hence, since all state equations satisfy the initial condition $x^*(0,z) = x^j(0,z) = A_{\text{init}} z$, we can apply Gronwall's inequality for initial value problems [88, Theorem 2.8] to obtain

$$\left\|x^*(t,y,\xi) - x^j(t,y,\xi)\right\|_2 \leq \frac{C_T |T^* - T^j| + C_\theta \left\|\theta^* - \theta^j\right\|_2}{C_x} \left(e^{C_x t} - 1\right).$$

Thus, we can deduce the uniform convergence of x^j to x^* in $C^0([0,1] \times \mathcal{Y} \times \Xi)$ as $j \to \infty$, which implies $\lim_{j\to\infty} \mathrm{J}(T^j, \theta^j) = \mathrm{J}(T^*, \theta^*)$. □

6.3 Time-discretization

In this section, we propose a numerical time-discretization scheme for the mean-field optimal control problem discussed in the previous section. For an a-priori fixed number of iteration steps $S \in \mathbb{N}$, we propose a semi-implicit discretization of the state equation (6.4), which yields

$$x_{s+1} = x_s - \tfrac{T}{S} A^*(Ax_{s+1} - z) - \tfrac{T}{S} D_1 \mathrm{R}(x_s, \theta) \in \mathbb{R}^{nC} \tag{6.9}$$

for $s = 0, \dots, S-1$ and initial state $x_0 = A_{\text{init}} z \in \mathbb{R}^{nC}$. This equation is equivalent to $x_{s+1} = g(x_s, z, T, \theta)$ with

$$g(x,z,T,\theta) \coloneqq (\mathrm{Id} + \tfrac{T}{S} A^* A)^{-1}(x + \tfrac{T}{S}(A^* z - D_1 \mathrm{R}(x,\theta))).$$

We denote by $\widehat{x}_s(y,\xi,T,\theta)$ the state of this discretization at time s given the ground truth y, the additive noise ξ, the stopping time T and the parameters θ. This discretization is equivalent to proximal gradient descent, where a gradient step on the regularizer and a proximal step on the data term is performed. Note that the inverse involving the linear operator A can be solved efficiently in many applications. The smoothness of the regularizer and the compactness of $\mathcal{Y}$, Ξ, $[0, T_{\max}]$ and Θ directly imply that

$$\widehat{x} : \mathcal{Y} \times \Xi \times [0, T_{\max}] \times \Theta \to \mathfrak{X}^{S+1} \subset (\mathbb{R}^{nC})^{S+1}$$

for a *compact* and convex set $\mathfrak{X}$. Then, the *discretized mean-field optimal control problem* is given by

$$\inf_{T\in[0,T_{\max}],\theta\in\Theta} \mathrm{J}_S(T,\theta), \tag{6.10}$$

where the discrete cost functional reads as

$$\mathrm{J}_S(T,\theta) \coloneqq \mathbb{E}_{(y,\xi)\sim\mathcal{T}} \left[\ell(\widehat{x}_S(y,\xi,T,\theta) - y)\right].$$

Theorem 6.3.1 *The minimum in* (6.10) *is attained.*

Proof. Let $(T^j, \theta^j) \in [0, T_{\max}] \times \Theta$ be a minimizing sequence for J_S with an associated state $\widehat{x}^j \coloneqq \widehat{x}(\cdot,\cdot,T^j,\theta^j)$. As in the time-continuous case, the compactness of $[0, T_{\max}] \times$

Θ implies the existence of a subsequence (again not relabeled) such that $(T^j, \theta^j) \to (T^*, \theta^*) \in [0, T_{\max}] \times \Theta$, where the associated state is given by $\widehat{x}^* := \widehat{x}(\cdot, \cdot, T^*, \theta^*)$. Then, we can estimate for any $(y, \xi) \in \mathcal{Y} \times \Xi$ and $s = 0, \dots, S-1$ as follows:

$$\left\| \widehat{x}^*_{s+1}(y,\xi) - \widehat{x}^j_{s+1}(y,\xi) \right\|_2 \leq C_T |T^* - T^j| + C_\theta \left\| \theta^* - \theta^j \right\|_2 + C_x \left\| \widehat{x}^*_s(y,\xi) - \widehat{x}^j_s(y,\xi) \right\|_2 .$$

Note that the constants C_T, C_θ and C_x depend on A, S, L_x, L_θ, $T_{\max}$, Θ and $\mathfrak{X}$. An induction argument reveals

$$\left\| \widehat{x}^*_{s+1}(y,\xi) - \widehat{x}^j_{s+1}(y,\xi) \right\|_2 \leq (C_T |T^* - T^j| + C_\theta \left\| \theta^* - \theta^j \right\|_2) \tfrac{1 - C_x^{s+1}}{1 - C_x}.$$

In particular, $\left\| x^*_S - x^j_S \right\|_{C^0(\mathcal{Y} \times \Xi)} \to 0$ as $j \to \infty$, which implies $\lim_{j \to \infty} \mathrm{J}_S(T^j, \theta^j) = \mathrm{J}_S(T^*, \theta^*)$. □

The existence of the discrete adjoint state is discussed in the subsequent theorem:

Theorem 6.3.2 *Let $(T^*, \theta^*) \in [0, T_{\max}] \times \Theta$ be a pair of control parameters for J_S and $x^* \in \mathcal{L} := L^2(\mathcal{Y} \times \Xi, (\mathbb{R}^{nC})^{S+1})$ the corresponding state. Then there exists a discrete adjoint state $p^* \in \mathcal{L}$ given by*

$$p^*_s(y,\xi) = (\mathrm{Id} - \tfrac{T^*}{S} D_1^2 \mathrm{R}(x^*_s(y,\xi), \theta^*))(\mathrm{Id} + \tfrac{T^*}{S} A^* A)^{-1} p^*_{s+1}(y,\xi) \tag{6.11}$$

for $s = S-1, \dots, 0$ with terminal condition $p^*_S(y,\xi) = -D\ell(x^*_S(y,\xi) - y)$.

Proof. First, we define the functional $G : \mathcal{L} \times [0, T_{\max}] \times \Theta \to \mathcal{L}$ representing the constraints as follows:

$$G(x, T, \theta)(y,\xi) = \begin{pmatrix} x_0(y,\xi) - A_{\mathrm{init}}(Ay + \xi) \\ x_1(y,\xi) - g(x_0(y,\xi), Ay + \xi, T, \theta) \\ \vdots \\ x_S(y,\xi) - g(x_{S-1}(y,\xi), Ay + \xi, T, \theta) \end{pmatrix}.$$

Then, the Lagrange functional $\mathrm{L} : \mathcal{L} \times [0, T_{\max}] \times \Theta \times \mathcal{L} \to \mathbb{R}$ using $\mathcal{L}^* \cong \mathcal{L}$ (Theorem 2.1.11) is given by

$$\mathrm{L}(x, T, \theta, p) := \mathbb{E}_{(y,\xi) \sim \mathcal{T}} \left[\ell(x_S(y,\xi) - y) + \sum_{s=0}^{S} \left\langle p_s(y,\xi), G_s(x(y,\xi), T, \theta) \right\rangle \right].$$

Following [248, Theorem 43.D], the Lagrange multiplier $p^* \in \mathcal{L}$ associated with (x^*, T^*, θ^*) exists if ℓ and G are (continuously) Frechét differentiable and the gradient of G is surjective. The differentiability requirements are immediately implied by the smoothness requirements of R. To prove the surjectivity of $D_1 G$, we first compute for $x \in \mathcal{L}$

$$D_1 G(x^*, T^*, \theta^*)(x)(y,\xi) = \begin{pmatrix} x_0(y,\xi) \\ x_1(y,\xi) - D_1 g(x^*_0(y,\xi), Ay + \xi, T^*, \theta^*) x_0(y,\xi) \\ \vdots \\ x_S(y,\xi) - D_1 g(x^*_{S-1}(y,\xi), Ay + \xi, T^*, \theta^*) x_{S-1}(y,\xi) \end{pmatrix}.$$

Thus, for any $w \in \mathcal{L}$ the solution $x \in \mathcal{L}$ of the equation $D_1 G(x^*, T^*, \theta^*)(x) = w$ is given by

$$\begin{aligned} x_0(y,\xi) &= w_0(y,\xi), \\ x_s(y,\xi) &= w_s(y,\xi) + D_1 g(x^*_{s-1}(y,\xi), Ay + \xi, T^*, \theta^*) x_{s-1}(y,\xi) \end{aligned}$$

for $s = 1, \dots, S$, which proves the surjectivity and thus the existence of Lagrange multipliers. Finally, (6.11) is implied by the optimality of L w.r.t. x. □

Next, we derive an optimality condition for the stopping time, which can easily be evaluated numerically.

Theorem 6.3.3 *Let (T^*, θ^*) be a stationary point of J_S with associated state x^* and adjoint state p^* as in Theorem 6.3.2. Then,*

$$\mathbb{E}_{(y,\xi)\sim\mathcal{T}}\left[\sum_{s=0}^{S-1}\left\langle p_{s+1}^*(y,\xi), (\mathrm{Id}+\tfrac{T^*}{S}A^*A)^{-1}(x_{s+1}^*(y,\xi)-x_s^*(y,\xi))\right\rangle\right] = 0. \tag{6.12}$$

Proof. Let us define $B(T) \coloneqq \mathrm{Id} + \frac{T}{S}A^*A$ and observe that

$$\tfrac{\mathrm{d}}{\mathrm{d}T}(B(T)^{-1}) = -B(T)^{-1}\left(\tfrac{\mathrm{d}}{\mathrm{d}T}B(T)\right)B(T)^{-1}.$$

The derivative of g w.r.t. T reads as

$$\begin{aligned}
&\tfrac{\mathrm{d}}{\mathrm{d}T}g(x,z,T,\theta)\\
&= -B(T)^{-1}\left(\tfrac{1}{S}A^*AB(T)^{-1}(x+\tfrac{T}{S}(A^*z-D_1\mathrm{R}(x,\theta)))-\tfrac{1}{S}(A^*z-D_1\mathrm{R}(x,\theta))\right)\\
&= -\tfrac{1}{T}B(T)^{-1}\left(x-B(T)^{-1}(x+\tfrac{T}{S}(A^*z-D_1\mathrm{R}(x,\theta)))\right).
\end{aligned}$$

Due to (6.9) the following relation holds true for the optimal $x^* \in \mathcal{L}$ and $s = 0, \dots, S-1$:

$$B(T)x_{s+1}^* = x_s^* + \tfrac{T}{S}(A^*z - D_1\mathrm{R}(x_s^*,\theta)).$$

Hence, the optimality condition of L w.r.t. T^* reads as

$$\mathbb{E}_{(y,\xi)\sim\mathcal{T}}\left[-\frac{1}{T^*}\sum_{s=0}^{S-1}\left\langle p_{s+1}^*(y,\xi), (\mathrm{Id}+\tfrac{T^*}{S}A^*A)^{-1}(x_{s+1}^*(y,\xi)-x_s^*(y,\xi))\right\rangle\right] = 0,$$

which proves this theorem. □

In fact, the optimality condition of the stopping time (6.12) is fulfilled if the velocity (difference of the states) is orthogonal to the adjoint states in the metric induced by the semi-implicit discretization along the trajectory across the dataset. Further, as we have derived in the proof, the first-order optimality condition is essentially the gradient of the cost functional and thus can be efficiently computed using backpropagation.

6.4 Stability Analysis

Here, we examine the stability of the proposed method, which quantifies the changes in the output caused by local perturbations of the observations and training parameters, respectively. The central assumption in both cases is that the distribution of the test data coincides with the distribution of the training data in the mean-field setting.

6.4.1 Stability Analysis w.r.t. Input

In what follows, we perform a stability analysis for the proposed algorithm, in which we derive upper bounds along the trajectories for different noise instances in the mean-field context. To this end, we first compute quantiles of the Lipschitz constant of the explicit update for the proposed discretization scheme given the

data distribution $\mathcal{T}$. Then, upper bounds for the difference of trajectories associated with one ground truth image and different noise instances drawn from the data distribution are derived using a recursion argument.

Let $x, \widetilde{x} \in \mathbb{R}^{nC}$, $T \in [0, T_{\max}]$, and $\theta \in \Theta$. We define the local Lipschitz constant of the explicit update step $x \mapsto x - \frac{T}{S} D_1 \mathrm{R}(x, \theta)$ as

$$L_x(x, \widetilde{x}, T, \theta) \coloneqq \frac{\left\| x - \frac{T}{S} D_1 \mathrm{R}(x, \theta) - \widetilde{x} + \frac{T}{S} D_1 \mathrm{R}(\widetilde{x}, \theta) \right\|_2}{\|x - \widetilde{x}\|_2},$$

where we set $L_x = 0$ if the denominator vanishes. Then, the cumulative distribution function F_S of the local Lipschitz constant on the data distribution $\mathcal{T}$ for $L \in \mathbb{R}$ is defined as

$$F_S(L) = \mathbb{P}\left(\max_{s=0,\dots,S} L_x(\widehat{x}_s(y, \xi, T, \theta), \widehat{x}_s(y, \widetilde{\xi}, T, \theta), T, \theta) \leq L : y \sim \mathcal{T}_{\mathcal{Y}},\ \xi, \widetilde{\xi} \sim \mathcal{T}_{\Xi} \right).$$

Thus, the maximum local Lipschitz constant of the explicit update step along each trajectory is bounded by $F_S^{-1}(1 - \delta)$ with probability $1 - \delta$.

Theorem 6.4.1 (Stability w.r.t. input) *Let $(T, \theta) \in [0, T_{\max}] \times \Theta$ be fixed control parameters, $y \sim \mathcal{T}_{\mathcal{Y}}$ and $\xi, \widetilde{\xi} \sim \mathcal{T}_{\Xi}$. We denote by $x, \widetilde{x} \in (\mathbb{R}^{nC})^{S+1}$ two solutions of the state equation associated with $z = Ay + \xi$ and $\widetilde{z} = Ay + \widetilde{\xi}$, and corresponding $x_0 = A_{\text{init}} z$ and $\widetilde{x}_0 = A_{\text{init}} \widetilde{z}$, respectively. The discrete state equations are given by*

$$x_{s+1} = g(x_s, z, T, \theta), \quad \widetilde{x}_{s+1} = g(\widetilde{x}_s, \widetilde{z}, T, \theta)$$

for $s = 0, \dots, S-1$. Let $\delta \in [0, 1)$,

$$\alpha_1(\delta) \coloneqq \left\| B^{-1} \right\|_2 F_S^{-1}(1 - \delta), \quad \beta_1 \coloneqq \tfrac{T}{S} \left\| B^{-1} \right\|_2 \|A\|_2$$

for $B \coloneqq \mathrm{Id} + \frac{T}{S} A^\top A$. Then,

$$\tfrac{1}{nC} \|x_{s+1} - \widetilde{x}_{s+1}\|_2 \leq \tfrac{1}{nC} \left(\alpha_1(\delta)^{s+1} \|A_{\text{init}}\|_2 + \tfrac{1 - \alpha_1(\delta)^{s+1}}{1 - \alpha_1(\delta)} \beta_1 \right) \|z - \widetilde{z}\|_2$$

holds true with probability $1 - \delta$.

Proof. The definition of the semi-implicit scheme (6.9) implies that for any $s = 0, \dots, S-1$ the inequality

$$\|x_{s+1} - \widetilde{x}_{s+1}\|_2 \leq \left\| B^{-1} \right\|_2 \left(F_S^{-1}(\delta) \|x_s - \widetilde{x}_s\|_2 + \tfrac{T}{S} \|A\|_2 \|z - \widetilde{z}\|_2 \right)$$

holds true with probability $1 - \delta$. By taking into account a recursion argument, the geometric series formula $\sum_{i=0}^{n} q^i = \frac{1 - q^{n+1}}{1 - q}$, and the estimate $\|x_0 - \widetilde{x}_0\|_2 \leq \|A_{\text{init}}\|_2 \|z - \widetilde{z}\|_2$ we obtain the desired result. □

6.4.2 Stability Analysis w.r.t. Parameters

Next, we elaborate on the stability of the proposed approach w.r.t. variations of the learned parameters $\theta \in \Theta$. To this end, we estimate the local Lipschitz constants of the TDV regularizer w.r.t. both of its arguments in the mean-field setting to derive upper bounds along the trajectories emanating from the same initial state, but with different parameters θ and $\widetilde{\theta}$. In detail, the perturbed parameters $\widetilde{\theta}$ are drawn from a uniform distribution supported on a component-wise relative ϵ-ball around θ. A recursion argument involving the estimated Lipschitz constants results in computable upper bounds for the norm difference along trajectories associated with θ and $\widetilde{\theta}$.

Let $B_\epsilon(\theta)$ be the component-wise relative ϵ-ball around $\theta = (K, K^i_{j,k,1}, K^i_{j,k,2}, w) \in \Theta$ w.r.t the ℓ^∞-norm, i.e.

$$\begin{aligned} B_\epsilon(\theta) = \Big\{ \widetilde{\theta} = (\widetilde{K}, \widetilde{K}^i_{j,k,1}, \widetilde{K}^i_{j,k,2}, \widetilde{w}) \in \Theta : \\ \left\| \widetilde{K} - K \right\|_\infty \leq \epsilon \left\| K \right\|_\infty \\ \left\| \widetilde{K}^i_{j,k,1} - K^i_{j,k,1} \right\|_\infty \leq \epsilon \left\| K^i_{j,k,1} \right\|_\infty , \\ \left\| \widetilde{K}^i_{j,k,2} - K^i_{j,k,2} \right\|_\infty \leq \epsilon \left\| K^i_{j,k,2} \right\|_\infty , \\ \left\| \widetilde{w} - w \right\|_\infty \leq \epsilon \left\| w \right\|_\infty \Big\}. \end{aligned}$$

Further, we denote by $\mathrm{proj}_\Theta : \mathbb{R}^p \to \Theta$ the orthogonal projection onto Θ, and by $\mathcal{U}(S)$ the uniform distribution for any bounded set $S \subset \mathbb{R}^p$. Then, the cumulative distribution function $F_{S,x}$ of the local Lipschitz constant of the regularizer w.r.t. its first component is given as

$$\begin{aligned} F_{S,x}(L) = \mathbb{P}\Big(\max_{s=0,\dots,S} L_x(\widehat{x}_s(y,\xi,T,\theta), \widehat{x}_s(y,\xi,T,\widetilde{\theta}), T, \theta) \leq L : \\ (y,\xi) \sim \mathcal{T}, \widetilde{\theta} \sim \mathcal{U}(\mathrm{proj}_\Theta(B_\epsilon(\theta))) \Big) \end{aligned}$$

for $L \in \mathbb{R}$. Likewise, we define the local Lipschitz constant of TDV w.r.t. its second argument as

$$F_{S,\theta}(L) = \mathbb{P}\left(\max_{s=0,\dots,S} L_\theta(\widehat{x}_s(y,\xi,T,\widetilde{\theta}), \theta, \widetilde{\theta}) \leq L : (y,\xi) \sim \mathcal{T}, \widetilde{\theta} \sim \mathcal{U}(\mathrm{proj}_\Theta(B_\epsilon(\theta))) \right)$$

for $L \in \mathbb{R}$, where

$$L_\theta(x, \theta, \widetilde{\theta}) := \frac{\left\| D_1 \mathrm{R}(x,\theta) - D_1 \mathrm{R}(x,\widetilde{\theta}) \right\|_2}{\left\| \theta - \widetilde{\theta} \right\|_2}.$$

Taking into account the above definitions we can state the stability theorem w.r.t. the learned parameters as follows:

Theorem 6.4.2 (Stability w.r.t. parameters) *Let $T \in [0, T_{\max}]$, $\theta \in \Theta$ and $\widetilde{\theta} \sim \mathcal{U}(\mathrm{proj}_\Theta(B_\epsilon(\theta)))$. We denote by $z = Ay + \xi$ an observation associated with $(y,\xi) \sim \mathcal{T}$, and by $\{x_s\}_{s=0}^S, \{\widetilde{x}_s\}_{s=0}^S \in (\mathbb{R}^{nC})^{S+1}$ two states satisfying (6.9) with initial conditions $x_0 = \widetilde{x}_0 = A_{\mathrm{init}} z$ and control parameters (T,θ) and $(T,\widetilde{\theta})$, respectively. Then, the inequality*

$$\frac{1}{nC} \left\| x_{s+1} - \widetilde{x}_{s+1} \right\|_2 \leq \frac{1}{nC} \frac{1 - \alpha_2(\delta)^{s+1}}{1 - \alpha_2(\delta)} \beta_2(\delta) \left\| \theta - \widetilde{\theta} \right\|_2 \tag{6.13}$$

holds true with probability $1 - \delta$ for $\delta \in [0,1)$, where

$$\alpha_2(\delta) = \left\| B^{-1} \right\|_2 F^{-1}_{S,x}(1 - \tfrac{\delta}{2}), \quad \beta_2(\delta) = \left\| B^{-1} \right\|_2 \tfrac{T}{S} F^{-1}_{S,\theta}(1 - \tfrac{\delta}{2})$$

for $B := \mathrm{Id} + \frac{T}{S} A^\top A$.

Proof. Again, using the definition of g yields

$$\left\| x_{s+1} - \widetilde{x}_{s+1} \right\|_2 \leq \alpha_2(\delta) \left\| x_s - \widetilde{x}_s \right\|_2 + \beta_2(\delta) \left\| \theta - \widetilde{\theta} \right\|_2$$

with probability $1 - \delta$, where we used

$$\left\| \mathrm{R}(x_s, \theta) - \mathrm{R}(\widetilde{x}_s, \widetilde{\theta}) \right\|_2 \leq F_{S,x}^{-1}(1 - \tfrac{\delta}{2}) \left\| x_s - \widetilde{x}_s \right\|_2 + F_{S,\theta}^{-1}(1 - \tfrac{\delta}{2}) \left\| \theta - \widetilde{\theta} \right\|_2 .$$

By exploiting a recursion argument and noting that the initial states coincide the theorem follows. □

Hence, this theorem provides a computable upper bound for the norm difference of two states w.r.t. perturbations of the TDV parameters. In particular, if (T, θ) is a local minimizer of the cost functional (6.10), then the stability analysis quantifies the robustness of the trajectories.

6.5 Numerical Results

In this section, we present numerical results for additive Gaussian denoising, medical image reconstruction, and single image super-resolution. To get an intuition for the local behavior of the learned regularizers, we pursue a nonlinear eigenfunction analysis. Moreover, we perform a stability analysis including adversarial attacks and worst-case generalization error estimates to demonstrate the robustness of the proposed method.

6.5.1 Training Details

In all experiments, we use the BSDS400 dataset [215] for training, which determines the discrete probability measure according to Remark 6.2.1. Thus, the control parameters (T, θ) are computed by minimizing the *discretized sampled optimal control problem*

$$\min_{T \in [0, T_{\max}], \theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} \ell(\widehat{x}_S(y^i, \xi^i, T, \theta) - y^i),$$

where $\ell(x) = \|x\|_2^2$ for Gaussian denoising and $\ell(x) = \sum_{i=1}^{nC} \sqrt{x_i^2 + \epsilon^2}$ for single image super-resolution with $\epsilon = 0.01$. We augment data of patch size 93×93 by randomly flipping the images horizontally or vertically, and by rotating the images by multiples of 90°. The ADAM optimizer [110] is employed with a mini batch size of 32 using 10^5 iterations, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, where the initial learning rate is $5 \cdot 10^{-3}$ for the FoE regularizer and $4 \cdot 10^{-4}$ for the TDV regularizer and the learning rate is halved every 25 000 iterations. The noise ξ is drawn randomly in each training iteration.

6.5.2 Additive Gaussian Denoising

As a first task, we consider additive Gaussian denoising implying $A = \mathrm{Id} \in \mathbb{R}^{nC \times nC}$, $\xi \sim \mathcal{N}(0, \sigma^2 \mathrm{Id})$ and $l = nC$ for $C = 1$ (gray-scale images) or $C = 3$ (color images). Thus, the data term specific for Gaussian image denoising is given by

$$\mathrm{D}(x, z) \coloneqq \left\| x - y \right\|_2^2 .$$

In the first experiments, we perform an ablation study of the number of blocks b, the number of scales a, and the potential function of the TDV regularizer. To this end, we evaluate the performance of the resulting TDV regularizers for additive gray-scale Gaussian denoising by computing the expected PSNR value on the BSDS68 dataset. Figure 6.3 depicts the expected PSNR values (top) and the optimal stopping times (bottom) as functions of the depth S for color-coded TDV regularizers with $a, b \in \{2, 3, 4\}$ and FoE_m^k regularizers with $m \in \{8, 24, 48\}$ filter kernels with

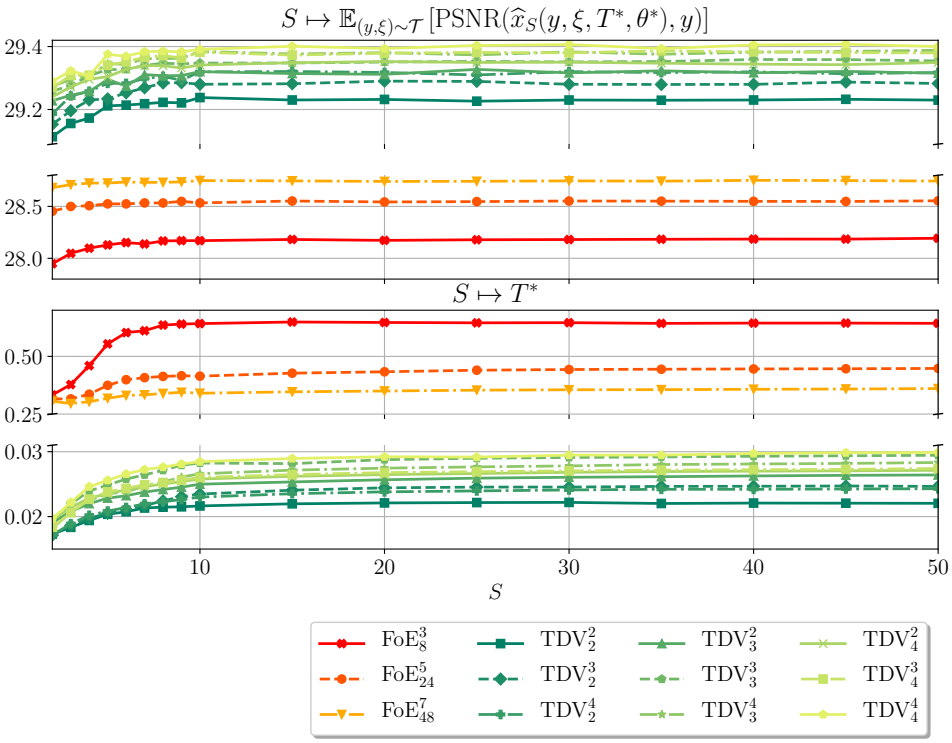


Figure 6.3: Expected PSNR value and optimal stopping time depending on S for various regularizers (gray-scale Gaussian denoising, $\sigma = 25$).

ψ	$\ln\cosh(x)$	$\frac{1}{2}\log(1+x^2)$	x
PSNR	29.3596	29.3662	29.3722

Table 6.1: Different possible choices for potential functions ψ of the TDV regularizer evaluated on gray-scale Gaussian denoising ($\sigma = 25$).

support $k \in \{3, 5, 7\}$. In all cases, the performance increases until $S \approx 10$, beyond this point the curves saturate. Thus, in all subsequent experiments, all regularizers are trained for $S = 10$. All considered variants of the TDV regularizer outperform the FoE regularizers in terms of expected PSNR score by a large margin, which originates from the ability of the TDV regularizer to combine image features in a nonlinear manner. Moreover, the expected PSNR values increase with the number of learnable parameters, which is correlated with the number of blocks and scales of the TDV regularizer and the number of features and the kernel size used in the FoE regularizer. However, beyond a certain complexity the performance increase saturates, that is why we use the TDV_3^3 and the FoE_{48}^7 regularizer in all further experiments. In addition, the optimal stopping time T^* also converges for all considered regularizers and is roughly 10-times higher for the FoE regularizer compared to the TDV regularizer. This effect is probably due to the larger receptive field of the TDV regularizer that enables a more effective processing of local information.

Table 6.1 lists the PSNR values for three possible choices of the potential functions ψ of the TDV regularizer. It turns out that the simplest potential function $\psi(x) = x$, which is neither bounded nor coercive, performs slightly better than the other potential functions. For this reason, we use $\psi(x) = x$ in all further experiments.

In what follows, we discuss the importance of the stopping time for the quality of the output image in analogy to the introductory example for the ROF model in Section 6.1. To this end, we plot the PSNR values of all BSDS68 test images and the corresponding expected PSNR value (red line) as a function of the stopping time (Figure 6.4) for the FoE (first row) and TDV regularizer (third row). In both plots, all curves approximately peak around the optimal stopping time, which is $T^* = 0.476$ for the FoE and $T^* = 0.017$ for the TDV regularizer. The optimal stopping time is also identified by the first-order condition of Theorem 6.3.3, which accounts for the zero-crossing of the gradient of the loss function w.r.t. the stopping time. This is visualized in Figure 6.4 for the FoE (second row) and the TDV (fourth row) regularizer. Further, to visually verify the importance of the proper choice of the optimal stopping time,

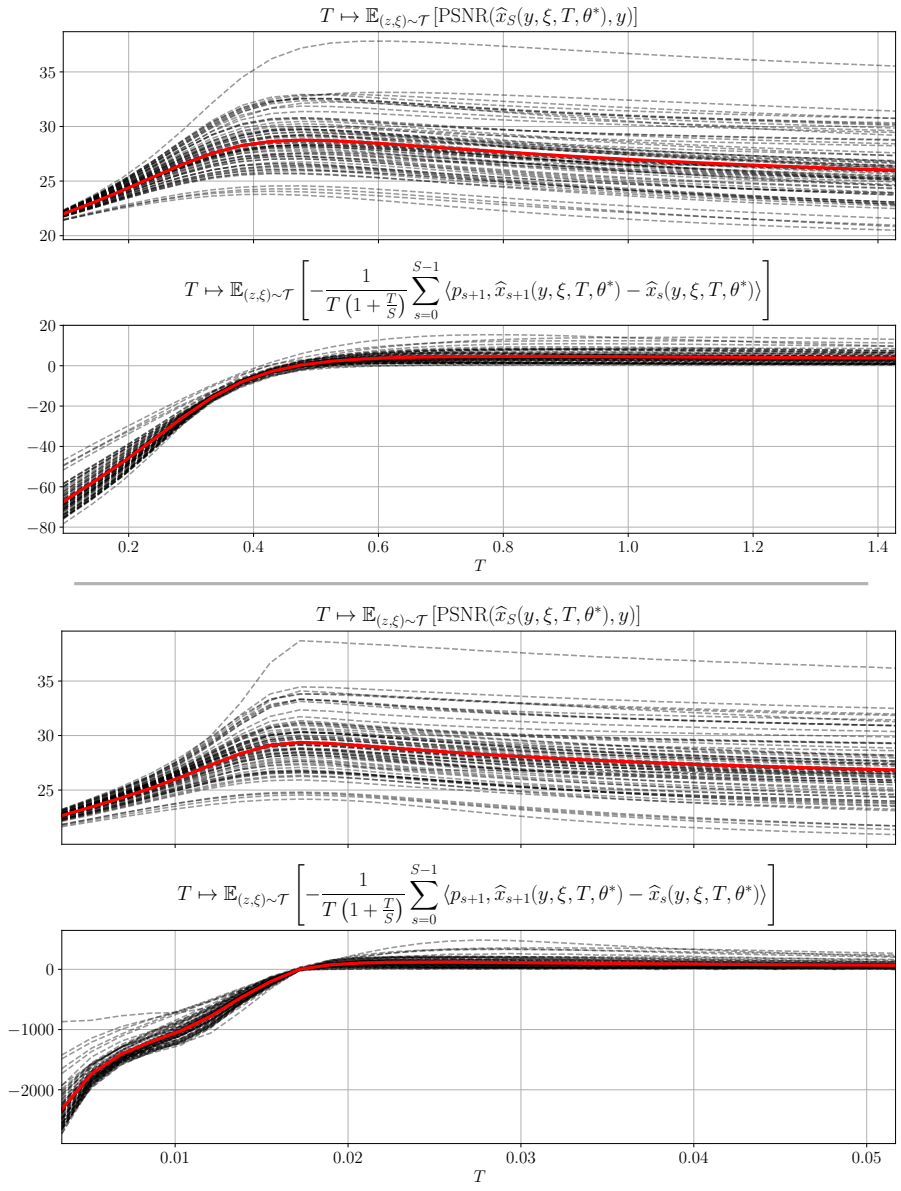


Figure 6.4: Depicted are the PSNR values over the BSDS68 dataset as a function of the stopping time T for the FoE^7_{48} (first row) and TDV^3_3 (third row) regularizer. The second and fourth row depict the associated first-order condition for the FoE^7_{48} and TDV^3_3 regularizer, respectively. Note that the red curves indicate the expected values of the corresponding quantity over the BSDS68 dataset, while the black dashed lines correspond to individual samples.

Figure 6.5 presents sequences of output images for gray-scale and color Gaussian denoising trained for $S = 10$. Starting from the noisy input image x_0 (second column), the noise level is gradually decreased until the output image x_{10} (fourth column) is obtained. Beyond this point, the algorithm generates over smoothed images, and details are lost. A qualitative comparison at the optimal stopping time T^* of the FoE and TDV regularizer in the sky/roof region of the water castle shows that the TDV regularizer generates images with sharp edges between homogeneously denoised regions, whereas the FoE regularizer generates ringing-like artifacts and smoother edges. With increasing stopping time $T > T^*$, the FoE regularizer smooths out image details such as the small windows and the chimney stripes in the zoom, while the prevailing structures of the image are preserved by the TDV regularizer. In addition, small image details such as the vertical ornaments above the large window in the zoom of the water castle are restored very well by the TDV regularizer compared to the FoE regularizer. Further, the TDV regularizer preserves these details much longer along the image trajectory.

Quantitative comparisons of expected PSNR values for additive gray-scale and color Gaussian denoising for $\sigma \in \{15, 25, 50\}$ on various image datasets are listed in Table 6.2 and Table 6.3. For $\mathrm{TDV}^3_{3,25}$, the PSNR values of our proposed TDV regularizer with three macro-blocks on three scales solely trained for $\sigma = 25$ are presented. Likewise, the $\mathrm{FoE}^7_{48,25}$ column lists the PSNR values obtained by applying

Figure 6.5: From left to right: Ground truth, noisy input with noise level $\sigma = 25$ and resulting output of the learned regularizers for $(S, T) \in \{(5, \frac{1}{2}T^*), (10, T^*), (15, \frac{3}{2}T^*), (20, 2T^*)\}$. First row: FoE^7_{48} for gray-scale Gaussian denoising. Second row: TDV^3_3 for gray-scale Gaussian denoising. Third row: FoE^7_{48} for color Gaussian denoising. Fourth row: TDV^3_3 for color Gaussian denoising. Note that the best images are framed in red and are obtained at the learned optimal stopping time T^*.

Table 6.2: Comparison of expected PSNR values for additive gray-scale Gaussian denoising for $\sigma \in \{15, 25, 50\}$ on various image datasets.

Dataset	σ	BM3D [73]	FoE_{48}^7	TNRD [55]	DnCNN [142]	FFDNet [250]	N^3Net [251]	FOCNet [249]	$\text{TDV}_{3,25}^3$	TDV_3^3
Set12	15	32.37	32.30	32.50	32.86	32.75	-	33.07	32.93	33.02
	25	29.97	29.79	30.05	30.44	30.43	30.55	30.73	30.68	30.68
	50	26.72	26.59	26.82	27.18	27.32	27.43	27.68	27.52	27.59
BSDS68	15	31.08	31.26	31.42	31.73	31.63	-	31.83	31.76	31.84
	25	28.57	28.75	28.92	29.23	29.19	29.30	29.38	29.37	29.37
	50	25.60	25.80	25.97	26.23	26.29	26.39	26.50	26.42	26.45
Urban100	15	32.34	31.62	31.98	32.67	32.43	-	33.15	32.66	32.91
	25	29.70	28.75	29.29	29.97	29.92	30.19	30.64	30.38	30.38
	50	25.94	25.18	25.71	26.28	26.52	26.82	27.40	26.94	27.04
# Parameters			3 842	26 645	555 200	484 800	705 895	53 513 120	387 394	387 394

Table 6.3: Comparison of expected PSNR values for additive color Gaussian denoising for $\sigma \in \{15, 25, 50\}$ on various image datasets.

Dataset	σ	BM3D [73]	$\text{FoE}_{48,25}^7$	CDnCNN [142]	FFDNet [250]	$\text{TDV}_{3,25}^3$
CBSDS68	15	33.52	33.53	33.89	33.87	34.12
	25	30.71	30.84	31.23	31.21	31.53
	50	27.38	27.43	27.92	27.96	28.26
Kodak24	15	34.28	34.30	34.48	34.63	35.01
	25	31.68	31.76	32.03	32.13	32.59
	50	28.46	28.42	28.85	28.98	29.44
McMaster	15	34.06	33.50	33.44	34.66	34.55
	25	31.66	31.33	31.51	32.35	32.47
	50	28.51	28.07	28.61	29.18	29.41
# Parameters			8 546	668 803	852 108	387 970

a FoE regularizer with 48 kernels of size 7×7 only trained for $\sigma = 25$. To apply the $\text{FoE}_{48,25}^7$ or $\text{TDV}_{3,25}^3$ model to different noise levels, we first rescale the noisy images $\overline{x}_{\text{init}} = \overline{z} = \frac{25}{\sigma} z$, then apply the learned scheme (6.9), and obtain the results via $x_S = \frac{\sigma}{25} \overline{x}_S$. In the fourth and last columns of Table 6.2, the PSNR values of the FoE and TDV regularizer–*individually* trained for each noise level–are listed. For color Gaussian denoising we only present results obtained by $\text{FoE}_{48,25}^7$ and $\text{TDV}_{3,25}^3$ to follow the evaluation standard of the related methods. We achieve state-of-the-art results for gray-scale and color image denoising with the TDV regularizer compared with models of similar complexity. Only FOCNet [249] performs slightly better for gray-scale images at the expense of more than a hundred times more trainable parameters. Additionally, the TDV regularizers yield higher PSNR values if their parameters are individually optimized for each noise level, as the comparison of the $\text{TDV}_{3,25}^3$ and TDV_3^3 columns shows. Note that the performance in terms of PSNR of the FoE_{48}^7 regularizer is rather good given the very small number of trainable parameters.

6.5.3 Two-dimensional Computed Tomography Reconstruction

To demonstrate the broad applicability of the proposed TDV regularizer, we perform a two-dimensional CT reconstruction using the $\text{TDV}_{3,25}^3$ regularizer, which was trained for gray-scale Gaussian image denoising and $S = 10$. We stress that the regularizer is applied *without* any additional training of the TDV parameters.

The task of CT is the reconstruction of an image given a set of projection measurements called sinogram, in which the detectors of the CT scanner measure the intensity of attenuated X-ray beams. Here, we use the linear attenuation model introduced in [252], where the attenuation is proportional to the intersection area of a triangle Γ_i, which is spanned by the X-ray source and a detector element as visualized in Figure 6.6, and the area of an image element. In detail, the sinogram z of an image x is computed by $z = A_R x$, where $A_R \in \mathbb{R}^{l \times n}$ is the lookup-table based area integral operator of [252] for R angles and 768 projections implying $l = 768 \cdot R$. Typically, a fully

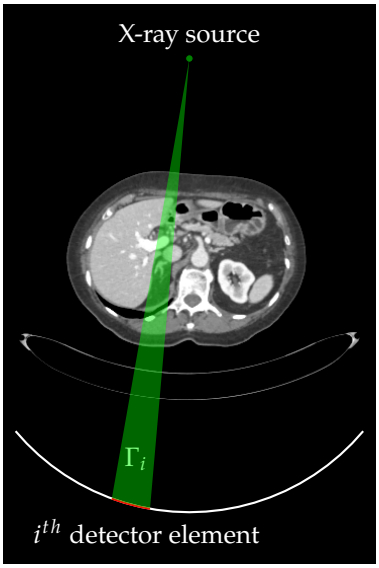


Figure 6.6: Illustration of the triangle Γ_i spanned by the X-ray source and the i^{th} detector element of the CT scanner. Note that the measured X-ray intensity at the i^{th} detector element is proportional to the matter in Γ_i.

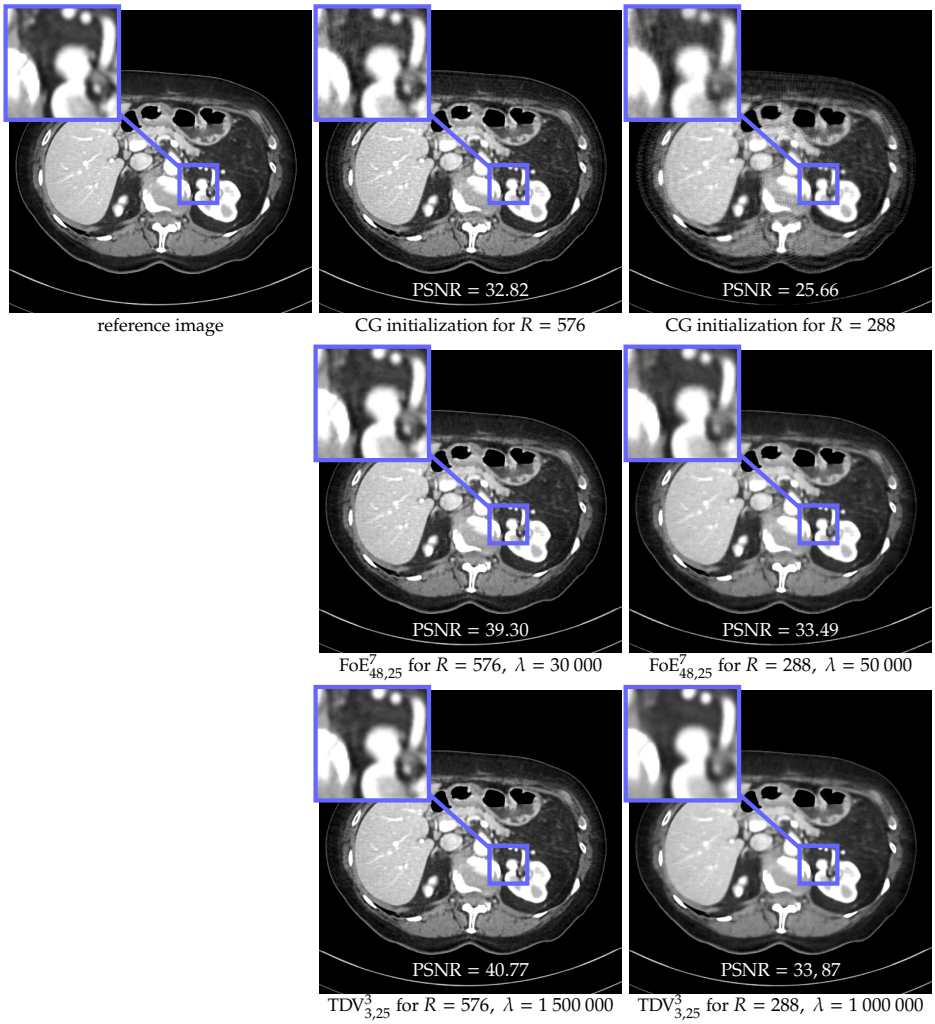


Figure 6.7: Fully sampled reference image (top left). Reconstruction results for 4-fold undersampling $R = 576$ (second column) and 8-fold undersampling $R = 288$ (third column). First row depicts the unregularized conjugate gradient reconstructions, second row the reconstruction results using the pretrained $\text{FoE}^7_{48,25}$ regularizer, and third row the reconstruction results using the pretrained $\text{TDV}^3_{3,25}$ regularizer.

sampled acquisition consists of 2304 angles. For this task, we consider the problem of angular undersampled CT [219], where only a fraction of the angles are measured. We use a 4-fold ($R = 576$) and 8-fold ($R = 288$) angular undersampling to reconstruct a representative image of the MAYO dataset [12] with $n = 768 \times 768$. To account for an imbalance of regularization and data fidelity, we manually scale the data fidelity term by $\lambda > 0$, i.e.

$$\mathrm{D}(x, z) := \frac{\lambda}{2} \|A_R x - z\|_2^2 .$$

The resulting smooth variational problem is optimized using accelerated gradient descent with Lipschitz backtracking with 1000 steps as discussed in [10].

We present qualitative and quantitative results for CT reconstruction in Figure 6.7 for a single abdominal CT image. The first row depicts from left to right the fully sampled reference image, an unregularized reconstruction for 4-fold angular undersampling ($R = 576$) and an unregularized reconstruction for 8-fold angular undersampling ($R = 288$), which were both computed by performing 50 steps of the conjugate gradient (CG) method on the data fidelity term. The corresponding reconstruction results by incorporating the $\text{FoE}^7_{48,25}$ or the TDV^3_3 regularizer are shown in the second and third row, respectively. Note that both parametric regularizers were solely trained for gray-scale Gaussian denoising and neither saw any medical images nor angular undersampling artifacts during training. Interestingly, both regularizers are able to suppress the undersampling artifacts while preserving, for instance, the fine vessels in the liver. For both angular undersampling patterns the reconstructions using the TDV^3_3 have a larger PSNR value compared to the FoE^7_{48} regularizer. Moreover, using the TDV^3_3 regularizer for $R = 288$, we are able to reconstruct a fine detail in the spine (highlighted in the zoom), which is not visible in the associated reconstruction

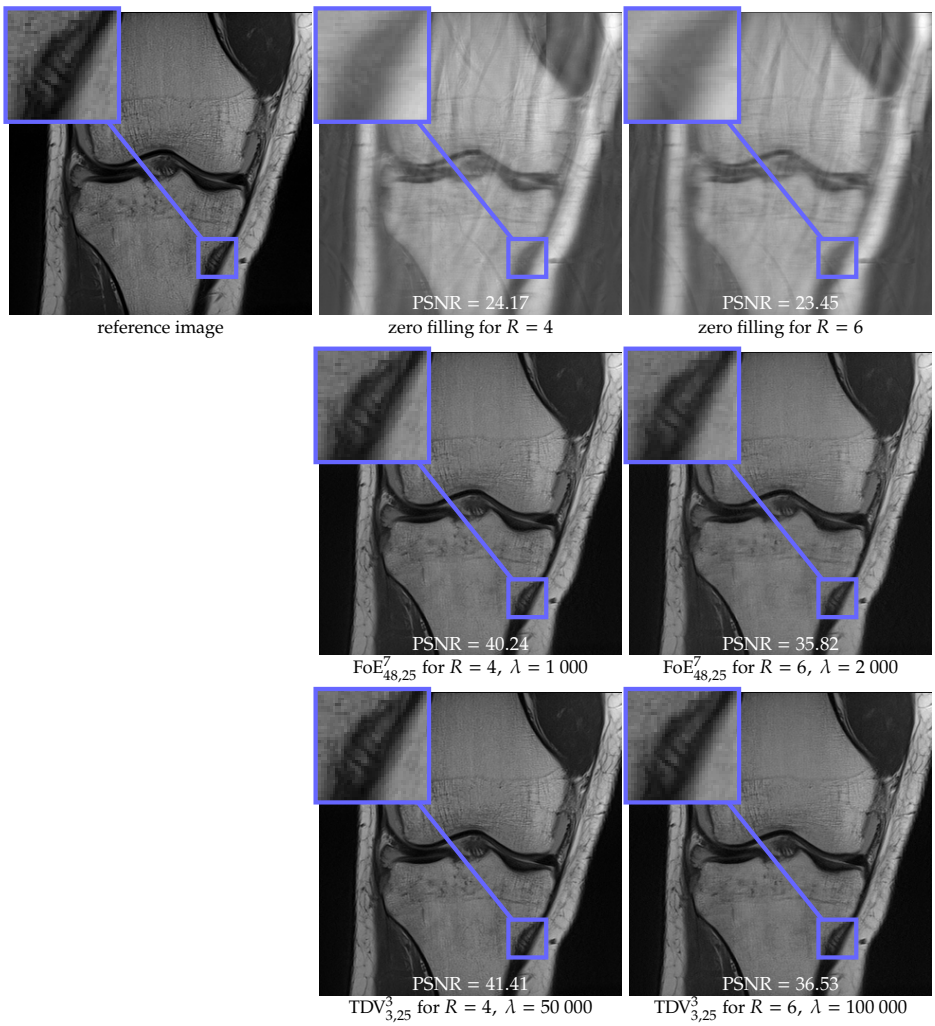


Figure 6.9: Reconstruction results for acceleration factor $R = 4$ (second column) and $R = 6$ (third column). The first row depicts from left to right the fully sampled reference image, the zero filling initialization for acceleration factors $R = 4$ and $R = 6$. The second row shows the reconstruction results using the pretrained $\mathrm{FoE}^7_{48,25}$ regularizer, and third row the reconstruction results using the pretrained $\mathrm{TDV}^3_{3,25}$ regularizer.

results using the FoE^7_{48} regularizer. This highlights that the learned regularizers can be effectively applied as a generic regularizer for linear inverse problems without any transfer learning, which is a particular benefit of the variational structure of the proposed approach.

6.5.4 Magnetic Resonance Imaging Reconstruction

To further point out the flexibility of the proposed approach, we apply our pretrained $\mathrm{FoE}^7_{48,25}$ and $\mathrm{TDV}^3_{3,25}$ regularizers, both learned for gray-scale Gaussian denoising and $S = 10$, to accelerated MRI *without* any further adaption of their parameters θ.

In accelerated MRI, k–space data $\{z_i\}_{i=1}^{N_C} \subset \mathbb{C}^n$ is acquired using N_C parallel coils, each measuring a fraction of the full k–space to reduce acquisition time [120]. Here, we use the data fidelity term

$$\mathrm{D}(x, \{z_i\}_{i=1}^{N_C}) = \frac{\lambda}{2} \sum_{i=1}^{N_C} \|M_R F C_i x - z_i\|_2^2 ,$$

where $\lambda > 0$ is a manually adjusted weighting parameter, $M_R \in \mathbb{R}^{n \times n}$ is a binary mask for R-fold Cartesian undersampling (see Figure 6.8), $F \in \mathbb{C}^{n \times n}$ is the discrete Fourier transform, and $C_i \in \mathbb{C}^{n \times n}$ are sensitivity maps. For further details, we refer the reader to [120]. We use 4-fold and 6-fold Cartesian undersampled MRI data to reconstruct a sample knee image. Again, we minimize the resulting variational energy by accelerated gradient descent with Lipschitz backtracking using 1000 steps [10].

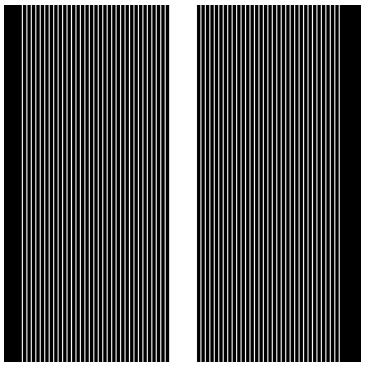


Figure 6.8: Two-dimensional Cartesian undersampling mask. The black rows are not acquired to accelerate the MRI acquisition process.

We perform an evaluation of the proposed approach on a representative slice of an undersampled MRI knee acquisition. The slice has a resolution of $n = 320 \times 320$ and $N_C = 15$ receiver coils were used during the acquisition. Figure 6.9 depicts qualitative results and PSNR values for the reconstruction of 4-fold and 6-fold undersampled k–space data. The first row shows the initial images obtained by applying the adjoint operator to the undersampled data. The second row depicts the reconstructed images using the FoE_{48}^7 regularizer and the third row the corresponding results obtained by $\text{TDV}_{3,25}^3$. Although both regularizers were not trained to account for undersampling artifacts, almost all artifacts are removed in the reconstructions and only some details are lost. As in the CT reconstruction task, using the TDV_3^3 regularizer results in a higher PSNR value of the reconstructions. In addition, the TDV_3^3 regularizer is capable of correctly reconstructing the three diagonal lines highlighted in the zoom, whereas the FoE_{48}^7 regularizer removes one line for the acceleration factor $R = 6$. This highlights the versatility and effectiveness of the proposed TDV regularizer since both CT and MRI reconstruction can be properly addressed *without* any fine-tuning of the learned parameters.

6.5.5 Single Image Super-resolution

In this subsection, we present numerical results for SISR. Here, the linear operator $A \in \mathbb{R}^{nC/\gamma^2 \times nC}$ is given as a downsampling operator, where $\gamma \in \{2, 3, 4\}$ denotes the scale factor. In detail, its adjoint operator coincides with MATLAB®'s bicubic upsampling operator `imresize`, which is an implementation of a scale factor-dependent bicubic interpolation convolution kernel in conjunction with a stride. Since this restoration problem substantially differs from Gaussian image denoising, the parameters of the TDV regularizer have to be optimized for this task individually.

Let nC be a multiple of γ^2 and $y \in \mathbb{R}^{nC}$ be a full resolution ground truth image patch uniformly drawn from the BSDS400 dataset. The observations

$$z = Ay + \xi \in \mathbb{R}^{nC/\gamma^2}$$

used for training are corrupted by additive Gaussian noise ξ with $\sigma \in \{0, 7.65\}$. For the initialization we set $A_{\text{init}} = \gamma A^\top$. The proximal map $(\text{Id} + \frac{T}{S} A^\top A)^{-1}$ is efficiently computed in Fourier space as advocated in [253]. Here, all results are obtained by training a TDV_3^3 regularizer for each scale factor individually.

We compare our SISR results with numerous state-of-the-art networks of similar complexity and list expected PSNR values of the Y-channel in the YCbCr color space over test datasets in Table 6.4. For the BSDS100 dataset, our proposed method using the TDV_3^3 regularizer achieves similar results as OISR-LF-s [254] with less than one third of the trainable parameters. Also the FoE_{48}^7 regularizer performs well given the limited number of learnable parameters. Figure 6.10 depicts a restored sequence of images for SISR with scale factor $\gamma \in \{2, 4\}$ using the FoE_{48}^7 and TDV_3^3 regularizers for two representative sample images of the Set14 dataset. The first row shows the image sequence using the FoE_{48}^7 regularizer and a scale factor $\gamma = 2$. The second row illustrates the corresponding image sequence using the TDV_3^3 regularizer. The resulting sequences for a scale factor $\gamma = 4$ are depicted in the third (FoE_{48}^7) and the fourth row (TDV_3^3). In all cases, the sequences start from the low-resolution initial image $x_0 = A_{\text{init}} z$, then interfaces are gradually sharpened and the best quality is achieved at the learned optimal stopping time $T = T^*$. Beyond this point, interfaces are artificially intensified. Also in the SISR task, the TDV_3^3 regularizer yields higher quality in terms of PSNR values at the optimal stopping time T^*. In addition, the images at the optimal stopping time generated by the TDV_3^3 regularizer have sharper interfaces. However, beyond the optimal stopping time T^* the TDV_3^3 regularizer over-intensifies the interfaces much more than the FoE_{48}^7 regularizer.

Figure 6.10: From left to right: Ground truth, low resolution initial image x_0 and resulting output of the learned regularizers for $(S, T) \in \{(5, \frac{1}{2}T^*), (10, T^*), (15, \frac{3}{2}T^*), (20, 2T^*)\}$. The first row shows the results using the FoE_{48}^7 regularizer for $\gamma = 2$ and the third row for $\gamma = 4$. The second row depicts the resulting trajectory using the TDV_3^3 regularizer for $\gamma = 2$ and the fourth row for $\gamma = 4$, respectively. Note that the best images are framed in red and are obtained at the learned optimal stopping time T^*.

Table 6.4: PSNR values of various state-of-the-art networks for single image super-resolution ($\sigma = 0$) with a comparable number of parameters.

Dataset	Scale	FoE_{48}^7	MemNet [255]	VDSR [256]	DnCNN-3 [142]	DRRN [257]	OISR-LF-s [254]	TDV_3^3
Set14	×2	32.77	33.28	33.03	33.03	33.23	33.62	33.35
	×3	29.63	30.00	29.77	29.81	29.96	30.35	29.94
	×4	27.89	28.26	28.01	28.04	28.21	28.63	28.41
BSDS100	×2	31.64	32.08	31.90	31.90	32.05	32.20	32.17
	×3	28.64	28.96	28.82	28.85	28.95	29.11	28.96
	×4	27.15	27.40	27.29	27.29	27.38	27.60	27.55
# Parameters		8 546	585 435	665 984	666 561	297 000	1 370 000	387 970

6.5.6 Eigenfunction Analysis

To get a better understanding of the local behavior of the FoE and the proposed TDV regularizer, we perform a nonlinear eigenfunction analysis [258]. To this end, we compute *nonlinear eigenfunctions* by minimizing the variational problem

$$\min_{x \in [0,1]^{nC}} \tfrac{1}{2} \left\| D_1 \mathrm{R}(x, \theta) - \Lambda(x) x \right\|_2^2, \tag{6.14}$$

where the generalized Rayleigh quotient defining the *eigenvalues* is given by

$$\Lambda(x) = \frac{\langle D_1 \mathrm{R}(x, \theta), x \rangle}{\|x\|_2^2}.$$

Note that (6.14) enforces $D_1 \mathrm{R}(x, \theta) \approx \Lambda(x) x$ for images with range space $[0, 1]$. We use Nesterov's projected accelerated gradient descent [98] to perform the optimization in (6.14). Due to the non-convexity of this minimization problem, the resulting eigenfunctions strongly depend on the initialization.

Figure 6.11 depicts six central image patches from the BSDS400 dataset (first row), which are used as the initialization, along with their eigenfunctions for gray-scale denoising (second row, $\sigma = 25$), color denoising (third row, $\sigma = 25$) and SISR (fourth row, $\gamma = 2$ and $\sigma = 0$). Note that each row shows the resulting eigenfunctions of the FoE_{48}^7 regularizer on the top and the corresponding eigenfunctions of the TDV_3^3 regularizer on the bottom. For denoising (second and third row), the generated eigenfunctions are composed of piecewise constant regions with smooth edges resulting in cartoon-like simplifications and contrast enhancement (see e.g. second/third column). The FoE_{48}^7 regularizer has a tendency to transform textured regions into blocky structures that are axis-aligned, whereas the TDV_3^3 regularizer transforms textured regions of the initial image into more complex repetitive structures such as stripes and dots. In contrast, the eigenfunctions for SISR (fourth row) exhibit fine-scaled texture details, which explain the property of the learned regularizers to recover high frequencies. Here, the FoE_{48}^7 regularizer also generates piecewise smooth regions and reduces contrast by favoring brown and green color shades throughout all examples, while the TDV_3^3 regularizer preserves the color to a large extent and adds high-frequency details as, for instance, can be seen at the noise and mouth region in the last column. These results clearly demonstrate that the learned regularizers are discriminative priors as they adapt to the specific image reconstruction tasks due to the discriminative learning approach.

6.5.7 Stability Analysis

In what follows, we elaborate on the stability of the proposed approach w.r.t. perturbations of the initial image and the learned parameters of the parametric regularizers. To this end, we numerically analyze the local structure of the regularization energy and experimentally validate Theorem 6.4.1 and Theorem 6.4.2. In all experiments in this subsection, we use the FoE_{48}^7 and TDV_3^3 regularizers. Let $x \in \mathbb{R}^{nC}$ be an image, $\xi \sim \mathcal{N}(0, \sigma^2 \mathrm{Id})$ Gaussian noise and θ parameters trained for gray-scale Gaussian denoising with $\sigma = 25$. Figure 6.12 visualizes the surface plots of the point-wise deep variation $[-1, 1] \ni (\zeta_1, \zeta_2) \mapsto \mathrm{r}(\zeta_1 x + \zeta_2 \xi, \theta)_i$ as a function of the contrast ζ_1 and the noise level ζ_2 for four prototypic pixels i marked in red. All surface plots exhibit distinct global minima and no high-frequency oscillations can be observed. Moreover, the point-wise deep variation strictly increases from the origin in all directions.

Motivated by the aforementioned surface plots, we can now conduct the stability analysis w.r.t. perturbations of the input, as illustrated in Figure 6.13. For this purpose, we estimate quantiles of the local Lipschitz constants L_x and L_θ of both regularizers by uniformly drawing 10^5 patches of size 128×128 from the BSDS400 dataset. To

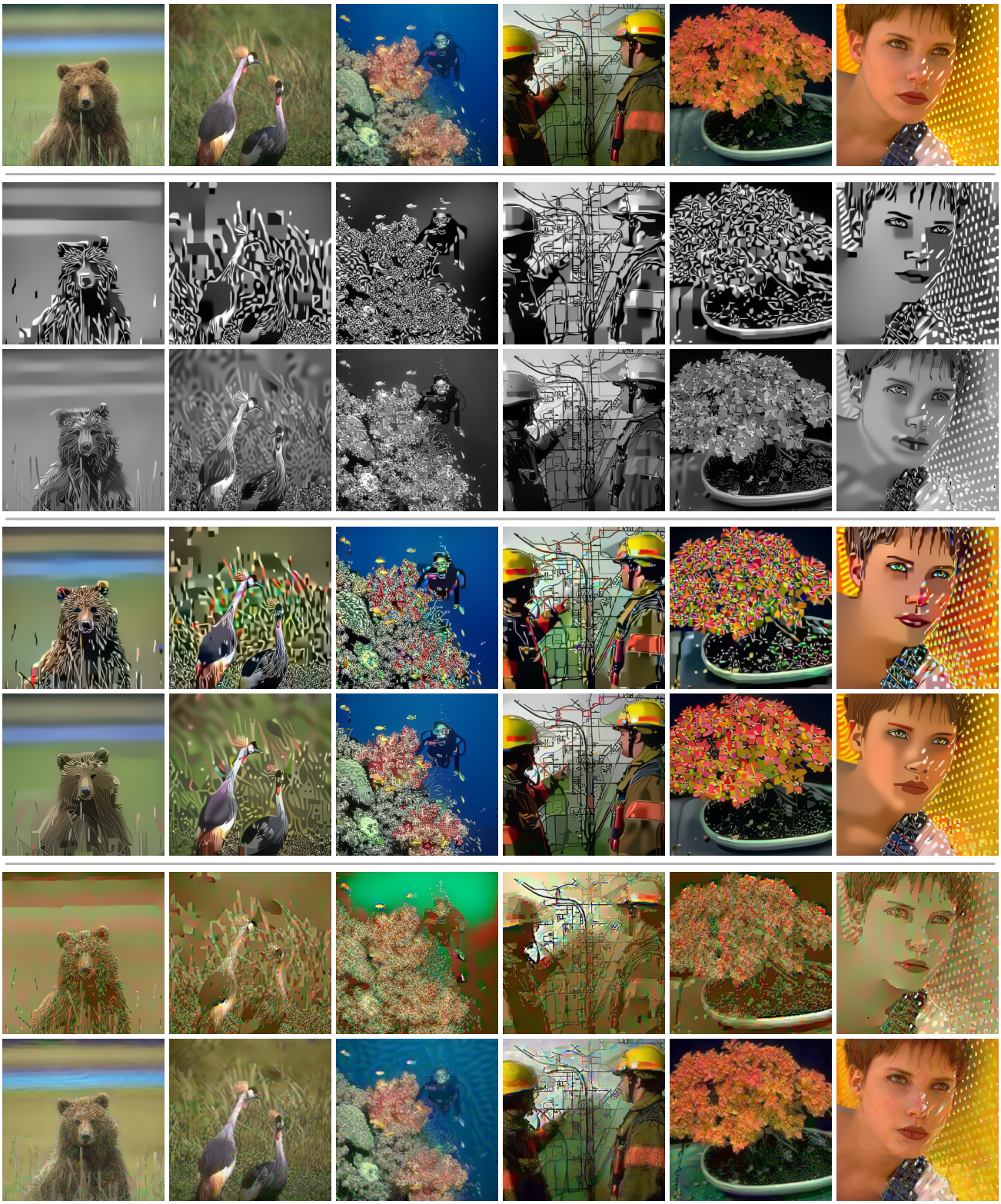


Figure 6.11: First row: initial images taken from BSDS400 dataset. Second row: eigenfunctions for gray-scale denoising ($\sigma = 25$) of the $\text{FoE}^7_{48,25}$ (top) and $\text{TDV}^3_{3,25}$ (bottom) regularizers. Third row: eigenfunctions for color denoising ($\sigma = 25$) of the $\text{FoE}^7_{48,25}$ (top) and $\text{TDV}^3_{3,25}$ (bottom) regularizers. Fourth row: eigenfunctions for single image super-resolution ($\gamma = 2,\ \sigma = 0$) of the FoE^7_{48} (top) and TDV^3_{3} (bottom) regularizers.

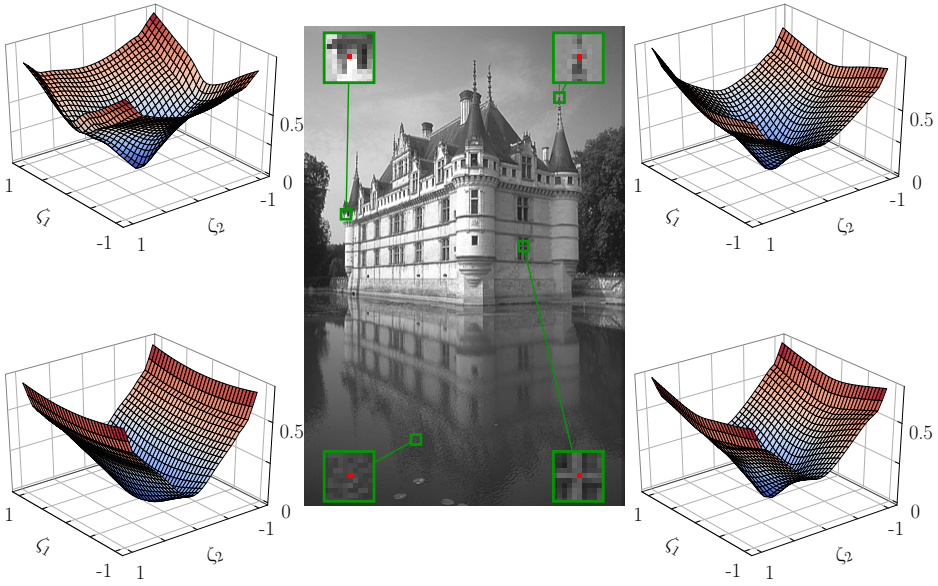


Figure 6.12: Surface plots of the pointwise deep variation $[-1,1] \ni (\zeta_1,\zeta_2) \mapsto \mathrm{r}(\zeta_1 x + \zeta_2 \xi, \theta)_i$ of four patches–each evaluated at the red center pixel.

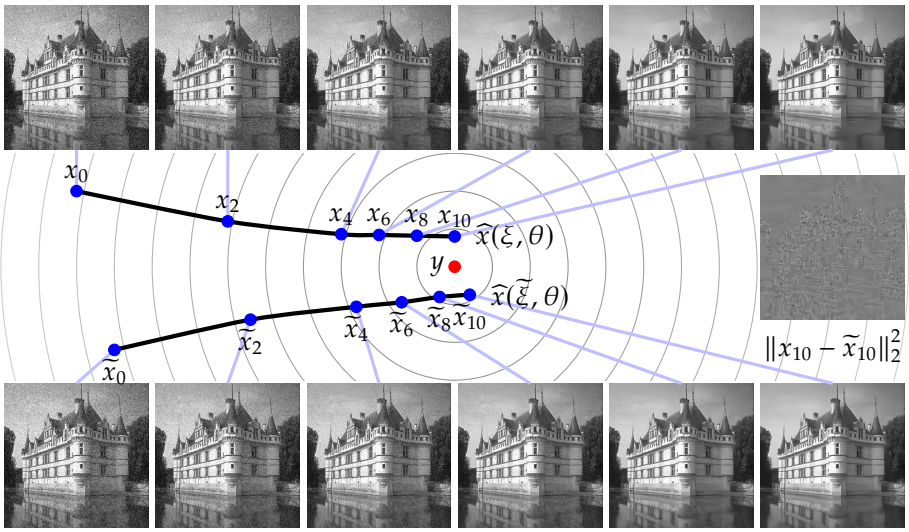


Figure 6.13: Graphical illustration of the effect of input (noise) variations for Gaussian gray-scale denoising. Given two noise samples $\xi, \widetilde{\xi}$ and the learned TDV_3^3 regularizer, both corresponding image trajectories originate from different initial states $x_0, \widetilde{x}_0$ and evolve towards the ground truth y during the denoising process. At the top and bottom the associated images of the TDV_3^3 regularizer are depicted and on the right the difference of the output images is shown in the interval $[-0.4, 0.4]$.

this end, we consider an image patch y randomly drawn from the BSDS68 dataset and let $\xi, \widetilde{\xi}$ be two noise instances independently drawn from $\mathcal{N}(0, \sigma^2 \mathrm{Id})$ for $\sigma = 25$. The associated observations are denoted by $z = y + \xi$ and $\widetilde{z} = y + \widetilde{\xi}$, resulting in the states x_s and $\widetilde{x}_s$. Figure 6.14 depicts the normalized norm differences $\frac{1}{nC} \|x_s - \widetilde{x}_s\|_2$ for all 68 patches (light blue curves), the corresponding mean curve (blue curve) as well as the upper bounds obtained from Theorem 6.4.1 for $\delta = 0.5$ (orange curve) and $\delta = 0.05$ (green curve) for gray-scale additive Gaussian denoising (first row) and single image super-resolution as a function of s for the FoE_{48}^7 (first column) and TDV_3^3 (second column) regularizer. It turns out that the normalized norm differences along the trajectories are only slightly overestimated for gray-scale Gaussian denoising and both considered regularizers, see the first row of Figure 6.14. Furthermore, the normalized norm differences strictly monotonically decrease for increasing s, which is also reflected in the upper bounds due to $\alpha_1(\delta) < 1$. In the case of single image super-resolution (second row of Figure 6.14), the bounds are less tight due to the inclusion of the non-trivial linear operators A with a non-empty nullspace. Furthermore, A_{init} with $\|A_{\text{init}}\|_2 = \gamma$ strongly influences the bounds. Note that the solutions of the single image super-resolution problem are not unique due to the structure of A. Nevertheless, the actual bandwidth of the normalized norm differences for samples of the BSDS68 dataset is rather small.

Next, we elaborate on the stability analysis w.r.t. variations of the learned parameters. Let y be a randomly drawn 128×128-patch from the BSDS68 dataset, which is

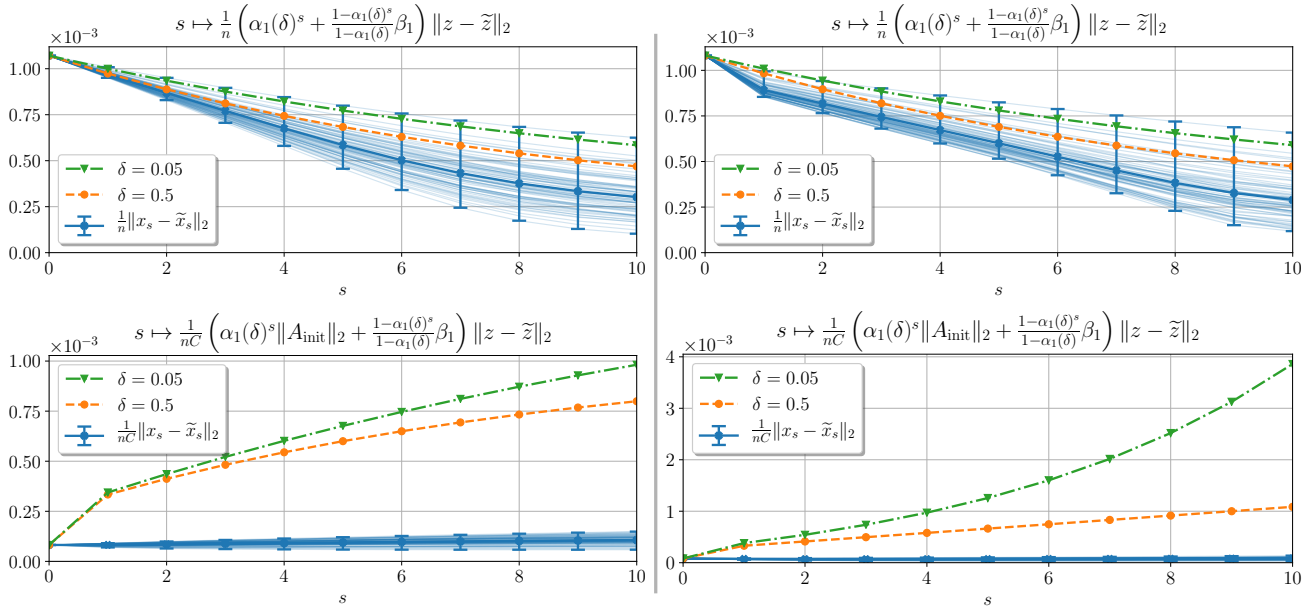


Figure 6.14: First row: stability analysis w.r.t. input variations of the proposed approach for FoE_{48}^7 (left) and TDV_3^3 (right) both trained for gray-scale Gaussian denoising with $\sigma = 25/S = 10$. Second row: stability analysis w.r.t. input variations for FoE_{48}^7 (left) and TDV_3^3 (right) both trained for SISR with $\gamma = 3/\sigma = 7.65/S = 10$.

corrupted by $z = Ay + \xi$. We consider parameters $\theta \in \Theta$ of each regularizer that are either optimized for gray-scale Gaussian denoising or single image super-resolution with $\sigma > 0$. The corrupted parameters $\widetilde{\theta}$ satisfy $\widetilde{\theta} \sim \mathcal{U}(\text{proj}_\Theta(B_\epsilon(\theta)))$ with $\epsilon = 0.1$. Hence, $\widetilde{\theta}$ is the element-wise sum of θ and strong uniform noise in the relative ϵ-ball around θ. We denote by x_s and $\widetilde{x}_s$ two states associated with θ and $\widetilde{\theta}$ emanating from the same noisy observation z, as illustrated in Figure 6.15. Thus, we expect that the normed difference of both trajectories increases with s. In Figure 6.16, the normalized norm differences of the states x_s and $\widetilde{x}_s$ for 68 random patches of the BSDS68 dataset (light blue curves), the corresponding mean curve (blue curve) as well as the theoretical upper bounds derived in Theorem 6.13 for $\delta = 0.5$ (orange curve) and $\delta = 0.05$ (green curve) are plotted as a function of s. The first row shows the results for gray-scale Gaussian denoising with $\sigma = 25$, while the second row shows the results for single image super-resolution for $\gamma = 3/\sigma = 7.65$. Further, the first column of Figure 6.16 presents the stability w.r.t. parameter variations of the FoE_{48}^7 regularizer and the second column the stability w.r.t. parameter variations of the TDV_3^3 regularizer. As expected, the normalized norm differences along the trajectories increase on average in all cases, which is also reflected in the derived upper bounds. The upper bounds for the FoE_{48}^7 regularizer on both considered tasks are rather tight due to the small number of trainable parameters. The upper bound of the TDV_3^3 regularizer for $\delta = 0.5$ and $s = 10$ is roughly four to five times higher than the expected curve of the normalized norm differences, which reflects the large variability of $\widetilde{\theta}$.

To conclude, in all cases the normalized norm differences (blue curves) are almost flat and the band width only slightly increases with s. This numerically validates that the proposed method is robust w.r.t. variations of both the input observations and the learned parameters.

6.5.8 Robustness against Adversarial Attacks

In what follows, we numerically check the robustness of the proposed method against adversarial attacks using the FoE_{48}^7 and the TDV_3^3 regularizer both trained for gray-scale Gaussian denoising with $\sigma = 25$. Let $y \in \mathbb{R}^{nC}$ be a ground truth image

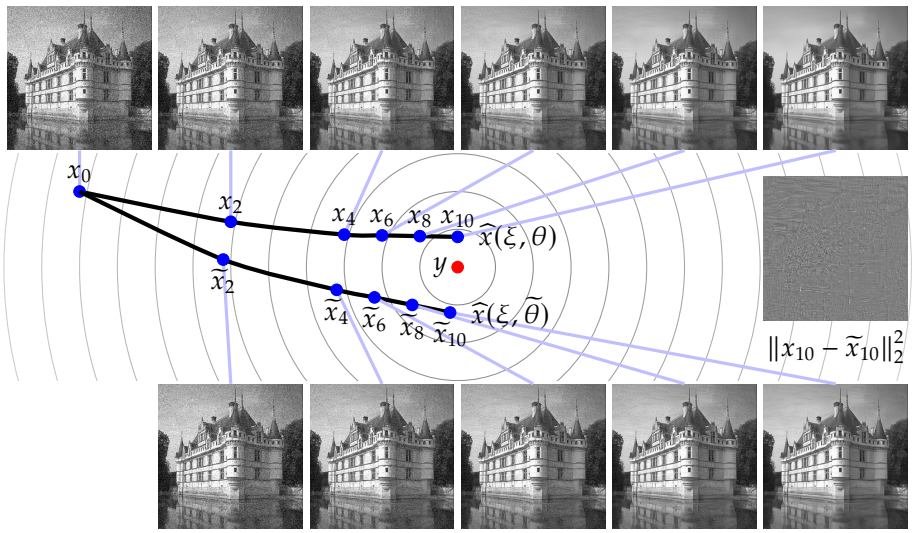


Figure 6.15: Graphical illustration of the effect of parameter variations for gray-scale Gaussian denoising. The state trajectories associated with θ and $\widetilde{\theta}$ originate from the same initial state x_0 and evolve towards the ground truth y. At the top and bottom the associated images of the TDV_3^3 regularizer are depicted and on the right the difference of the output images is shown in the interval $[-0.15, 0.15]$.

patch, $\xi \sim \mathcal{N}(0, \sigma^2 \,\text{Id})$ Gaussian noise and $z = y + \xi \in \mathbb{R}^{nC}$. The adversarial noise $\widetilde{\xi}$ for $\epsilon > 0$ is computed via

$$\max_{\widetilde{\xi} \in \mathbb{R}^l : \|\widetilde{\xi}\|_2 \leq \epsilon} \left\| \widehat{x}_S(y, \xi + \widetilde{\xi}, T, \theta) - y \right\|_2^2.$$

Thus, we seek the noise structure that leads to the largest deviation from y within an ϵ-ball around z.

Figure 6.17 shows two different ground truth image patches, the corresponding restored images using either the FoE_{48}^7 or the TDV_3^3 regularizer along with the computed regularizer-dependent adversarial noise structures and the corresponding output images for two radii $\epsilon \in \{1, 2\}$. As a result, with increasing radius ϵ high-frequency patterns are generated in the adversarial noise, which are emphasized in the corresponding output images. The adversarial noise associated with the FoE_{48}^7 regularizer consists of locally concentrated irregular stripe patterns, while the adversarial noise of the TDV_3^3 regularizer admits more regular stripe and dot patterns, which are typically difficult to reconstruct given just the noisy observation. In summary, in all adversarial attacks, no new structures are hallucinated by the regularizers, only existing patterns (either in the original image y or in the adversarial noise $\widetilde{\xi}$) are intensified in $x_S(\widetilde{\xi})$. Hence, the adversarial noise has only a local influence on the final reconstruction $x_S(\widetilde{\xi})$, which numerically validates that both the FoE_{48}^7 and the TDV_3^3 regularizer are stable against adversarial attacks.

6.5.9 Empirical Upper Bound for Generalization Error

Next, we experimentally compute worst-case upper bounds for the generalization error of the FoE_{48}^7 and the TDV_3^3 regularizer trained for gray-scale Gaussian denoising with $\sigma = 25$. As a starting point, let $\mathcal{Y} \subset [0, 1]^n$ be the set of natural images with distribution $\mathcal{T}_\mathcal{Y}$. Further, let $\mathcal{Y}' \subset \mathcal{Y}$ be a collection of 10^5 ground truth image patches of size 128×128 randomly drawn from the BSDS400 dataset, the BSDS68 dataset, and the DIV2K validation set [11]. The uniform distribution on $\mathcal{Y}'$ is denoted by $\mathcal{T}_{\mathcal{Y}'}$. Following [129], the *empirical risk* w.r.t. $\mathcal{Y}'$ is defined as

$$E_{\text{emp}}(\mathcal{Y}') := \frac{1}{|\mathcal{Y}'|} \sum_{y' \in \mathcal{Y}'} \ell(\widehat{x}_S(y', \xi_{y'}) - y'),$$

the *expected loss* of $\mathcal{Y}$ reads as

$$E(\mathcal{Y}) := \mathbb{E}_{y \sim \mathcal{T}_\mathcal{Y}} \ell(\widehat{x}_S(y, \xi_y) - y),$$

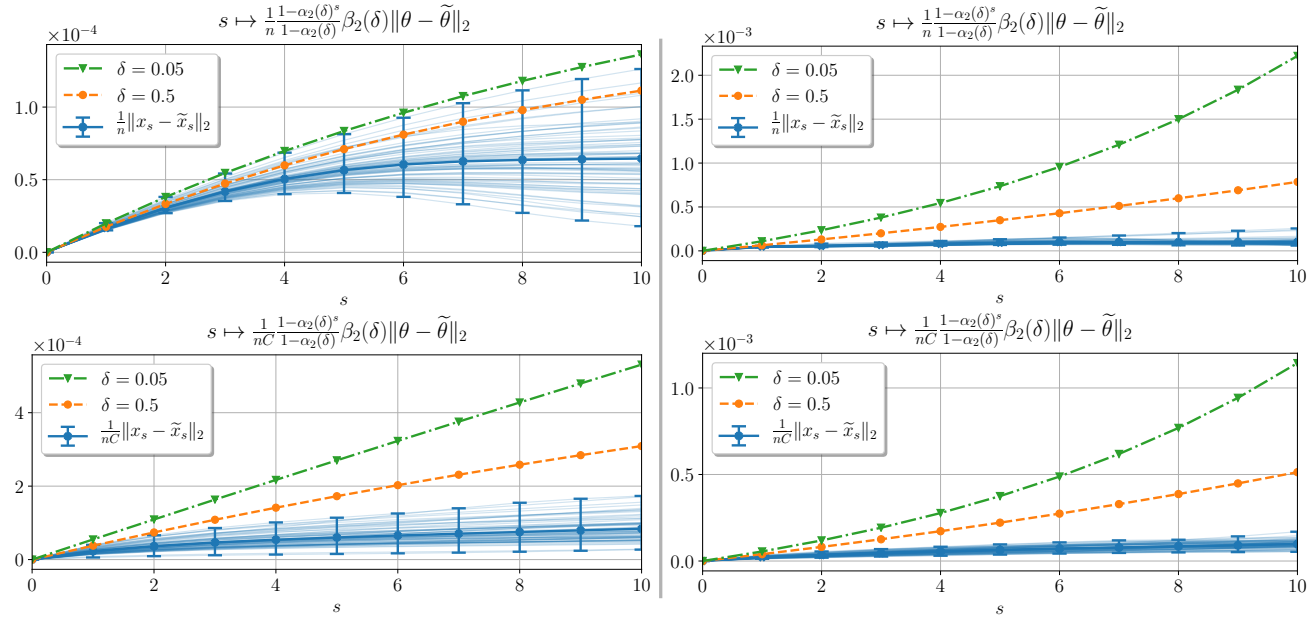


Figure 6.16: First row: stability analysis w.r.t. the learned parameters of FoE_{48}^7 (left) and TDV_3^3 (right) both trained for gray-scale Gaussian denoising with $\sigma = 25/S = 10$. Second row: stability analysis w.r.t. the learned parameters of FoE_{48}^7 (left) and TDV_3^3 (right) both trained for SISR with $\gamma = 3/\sigma = 7.65/S = 10$.

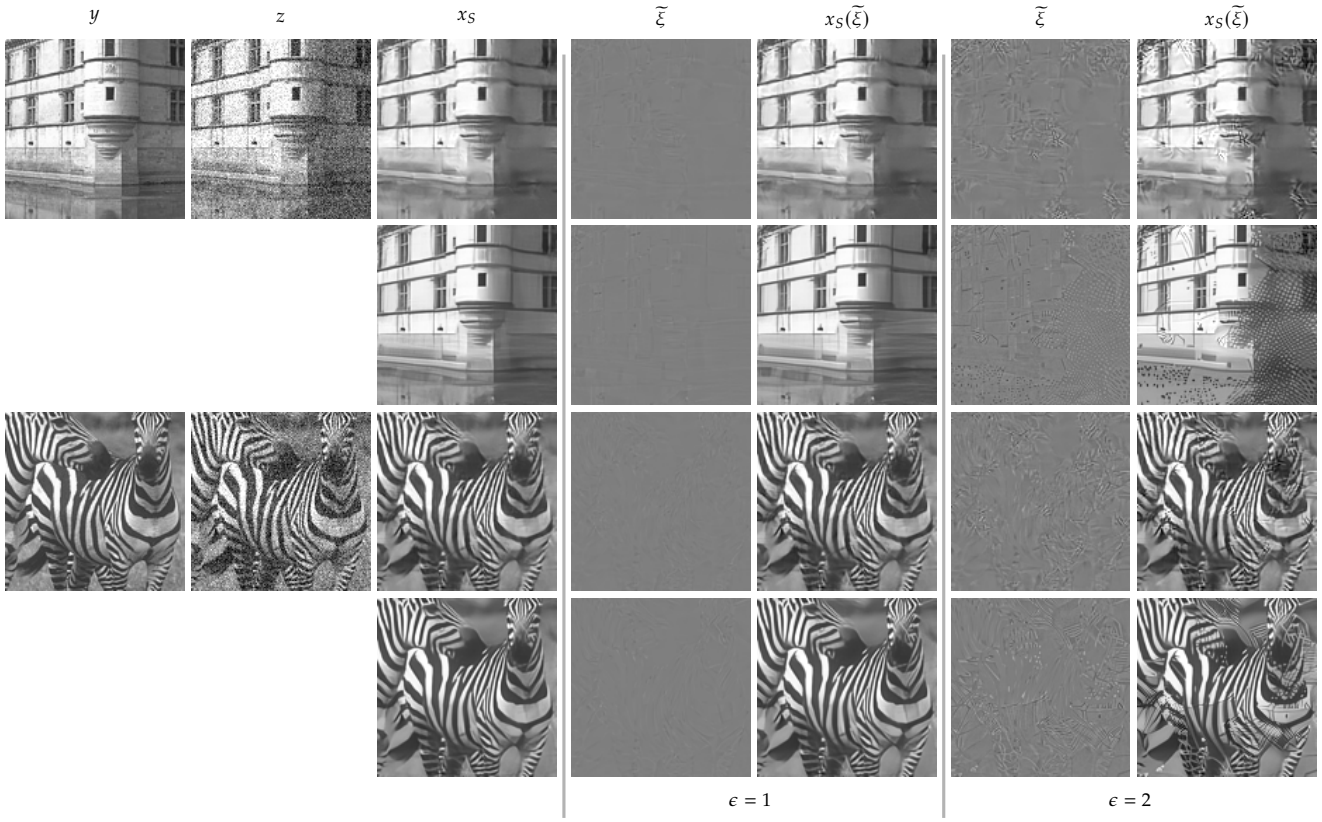


Figure 6.17: From left to right: ground truth image patch y (first column), noisy observation $z = y + \xi$ (second column), reconstructed image (third column), pairs of (adversarial) noise and resulting output for radii $\epsilon = 1$ (fourth/fifth column) and $\epsilon = 2$ (sixth/seventh column), where $x_S(\widetilde{\xi}) = \widehat{x}_S(y, \xi + \widetilde{\xi}, T, \theta)$. The first and third row depict results of the FoE_{48}^7 regularizer and the second and fourth row the corresponding results of the TDV_3^3 regularizer, which both were trained for gray-scale Gaussian denoising with $\sigma = 25/S = 10$. The adversarial noise $\widetilde{\xi}$ is displayed in the range $[-0.5, 0.5]$.

where the loss $\ell(x) = \|x\|_2^2$ is the quadratic ℓ^2-norm, ξ_y and $\xi_{y'}$ are a-priori sampled noise instances drawn from $\mathcal{N}(0, \sigma^2 \mathrm{Id})$, and $\widehat{x}_S(y, \xi) = \widehat{x}_S(y, \xi, T, \theta)$. In this case, the *generalization error* for $\mathcal{Y}'$ is defined as the absolute difference between the expected loss and the empirical loss, i.e.

$$|E(\mathcal{Y}) - E_{\mathrm{emp}}(\mathcal{Y}')|.$$

A worst-case upper bound for this generalization error is given by

$$\max_{\widetilde{y} \in [0,1]^n} \ell(\widehat{x}_S(\widetilde{y}, \xi) - \widetilde{y}) - E_{\mathrm{emp}}(\mathcal{Y}') \tag{6.15}$$

for an a-priori sampled $\xi \sim \mathcal{N}(0, \sigma^2 \mathrm{Id})$ since the expected loss is estimated from above by its single worst realization.

To analyze the dependency between the loss and the regularization energy, we show scatter plots of the plane spanned by $\ell(\widehat{x}_S(y', \xi_{y'}) - y')$ and $\mathrm{R}(y', \theta)$ in Figure 6.18 (first column) for all $y' \in \mathcal{Y}'$ and both regularizers. We observe a strikingly linear dependency, which is reflected by an R^2-value of 0.984 for the FoE_{48}^7 regularizer and 0.985 for the TDV_3^3 regularizer of a linear regression with intercept. This linear dependency gives rise to a probabilistic analysis of worst-case upper bounds for the generalization error on quantiles of the corresponding regularization energy. For this reason, we define the cumulative distribution function

$$F_{\mathrm{R}}(H) = \mathbb{P}\left(\mathrm{R}(y', \theta) \leq H : y' \sim \mathcal{T}_{\mathcal{Y}'}\right)$$

for $H \in \mathbb{R}$. Note that $F_{\mathrm{R}}^{-1}(q)$ for $q \in (0, 1]$ defines the q^{th}-quantile of the regularization energy over $\mathcal{Y}'$. Then, we derive an upper bound for the generalization error restricted to the subset

$$\mathcal{Y}_q' = \{y' \in \mathcal{Y}' : \mathrm{R}(y', \theta) \leq F_{\mathrm{R}}^{-1}(q)\}$$

for $q \in (0, 1]$. In this setting, the expected loss of the q^{th}-quantile is estimated from above by $\ell(\widehat{x}_S(\widetilde{y}_q) - \widetilde{y}_q)$, where

$$\widetilde{y}_q \in \operatorname*{argmax}_{y \in [0,1]^n} \ell(\widehat{x}_S(y, \xi) - y) \quad \text{s.t.} \quad \mathrm{R}(y) \leq F_{\mathrm{R}}^{-1}(q). \tag{6.16}$$

In detail, we try to identify the image patch $\widetilde{y}_q$ that leads to the worst-case loss ℓ among all image patches in $[0, 1]^n$ such that their regularization energy $\mathrm{R}(\widetilde{y}_q)$ is at most $F_{\mathrm{R}}^{-1}(q)$. Hence, an upper bound for the generalization error on the set $\mathcal{Y}_q'$ is given by

$$\mathrm{G}(q) \coloneqq \ell(\widehat{x}_S(\widetilde{y}_q, \xi) - \widetilde{y}_q) - E_{\mathrm{emp}}(\mathcal{Y}_q').$$

To compute the worst-case ground truth image, we account for the constraint in (6.16) by a quadratic barrier approach and solve the resulting minimization problem using Nesterov's accelerated gradient method [98] with Lipschitz backtracking starting from a patch with uniform noise.

Figure 6.18 (second column) depicts the semi-logarithmic plots of the upper bound for the expected loss (solid curve), the empirical risk (dashed curve), and the upper bound for the generalization error (dash-dotted curve) on the set $\mathcal{Y}_q'$ for both regularizers as a function of q. For convenience, the third plot in the top part of Figure 6.18 depicts the corresponding PSNR curves measured in decibels. It highlights that the worst-case generalization error is around 15dB for the FoE_{48}^7 regularizer and around 10dB for the TDV_3^3 regularizer on the considered dataset $\mathcal{Y}'$. We note that the upper bound for the generalization error slightly increases with larger regularization energy values for the TDV_3^3 regularizer represented by larger q. We observe that the upper bound is not tight, which originates from the minimization in (6.16) among all patches in $[0, 1]^n$. The computed worst-case patches $\widetilde{y}_q$ along with the reconstructed output images $\widehat{x}_S(\widetilde{y}_q, \xi)$ are shown in Figure 6.19 in the center part for the FoE_{48}^7 regularizer and at the bottom part for the TDV_3^3 regularizer. The worst-case ground truth patches

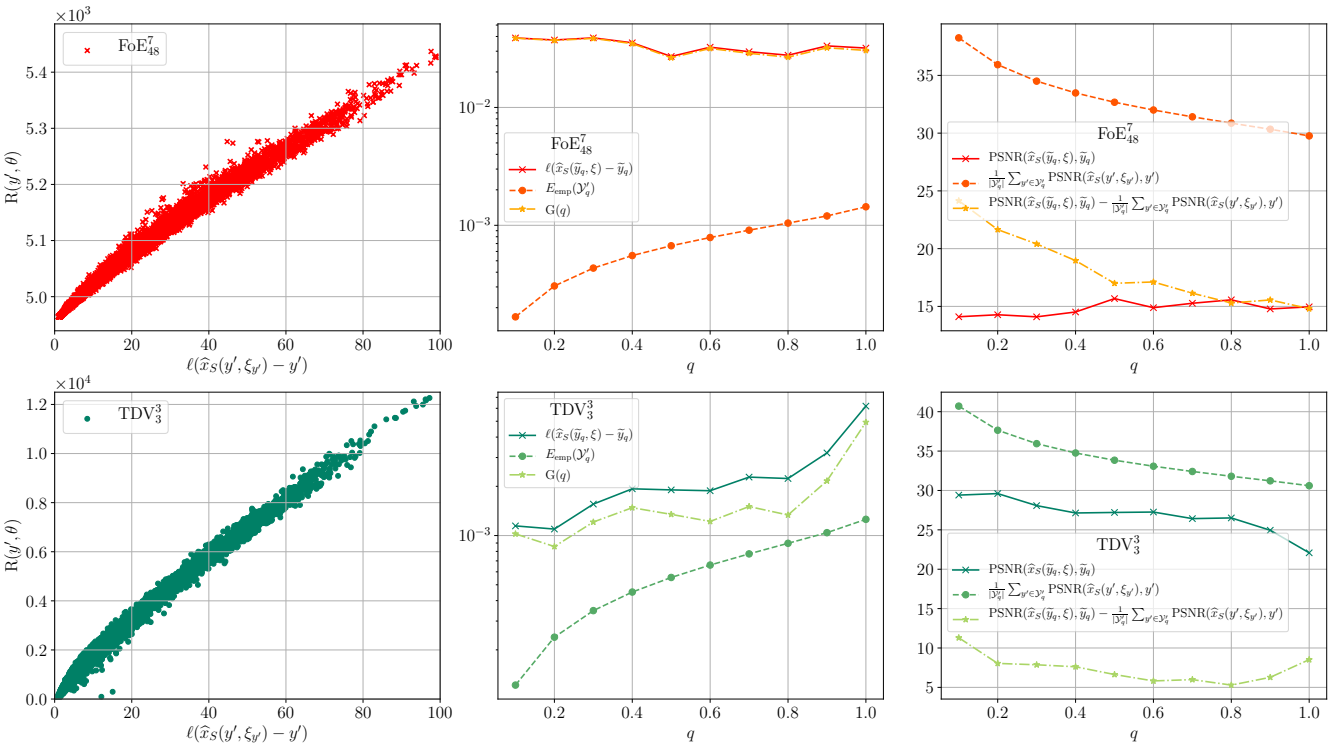


Figure 6.18: The first column depicts a scatter plot in the $\ell(\widehat{x}_S(y, \xi_y) - y)$ and $\mathrm{R}(y, \theta)$-plane for the FoE^7_{48} (first row) and TDV^3_3 (second row) regularizer. The second column shows the corresponding semi-logarithmic plots of the upper bound for the expected loss (solid curve), the empirical risk (dashed curve) and the upper bound for the generalization error (dashed dotted curve) restricted to $\mathcal{Y}'_q$ as a function of the quantiles q for both considered regularizers. The third column highlights the associated PSNR curves measured in decibels.

for the FoE^7_{48} regularizer basically consist of two noise patterns that differ in variance and partition the image into distinct regions. In the output image, the low variance noise pattern is smoothed out and the high variance patterns are smoothed and turned into a stripe pattern. With increasing q the fraction of the ground truth patches covered by the high variance pattern increases. In contrast, the worst-case ground truth patches for the TDV^3_3 regularizer consist of high-oscillatory stripe patterns and checkerboard artifacts for small q, whereas noise and texture patterns are dominant for higher values of q. We emphasize that all generated patches are artificial and not likely to be contained in any natural image. That is why the upper bound for the generalization error tends to overestimate the actual generalization error.

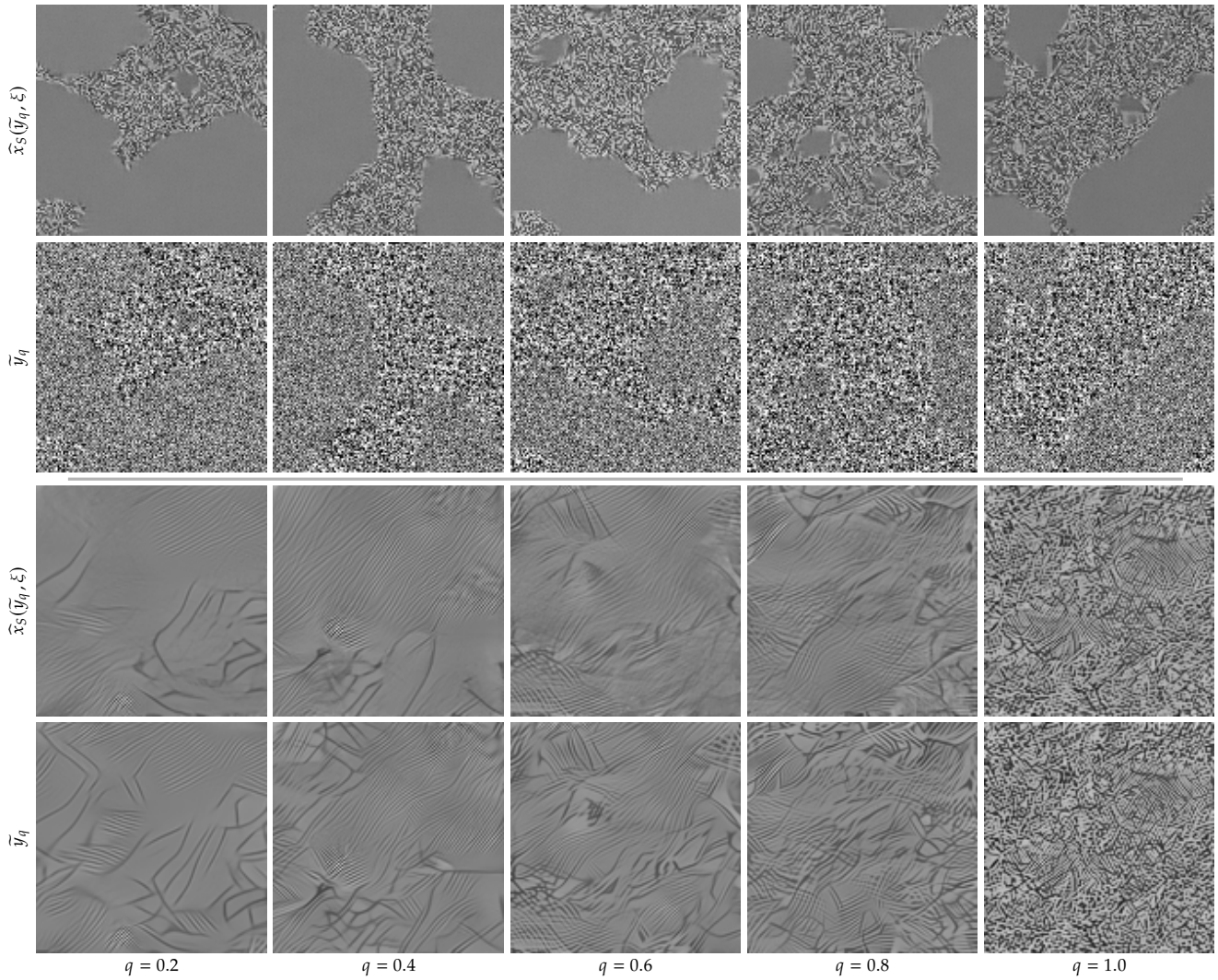


Figure 6.19: Pairs of worst-case ground truth $\widetilde{y}_q$ and corresponding output image $\widehat{x}_S(\widetilde{y}_q, \xi)$ of the FoE_{48}^7 regularizer (first and second row) and the TDV_3^3 (third and fourth row) regularizer.

6.6 Application to SparseCT Reconstruction

In this section, we elaborate on how to extend the TDV regularizer to 3-dimensional inverse problems such as SparseCT. First, we provide a brief overview of SparseCT. Then, the TDV regularizer is applied to this task such that the training is time- and memory-efficient. Finally, numerical experiments show the potential of the proposed approach.

SparseCT is a technique for CT dose reduction in helical CT acquisition that implements compressed sensing with a multislit collimator (MSC) (see Figure 6.20) for undersampled projection data acquisition and iterative reconstruction with a sparsity-enforcing regularizer [83]. The MSC is composed of several narrow slits along the slice dimension (z) and can move linearly to change the z-undersampling pattern along the gantry rotation, which improves incoherence for compressed sensing reconstruction [259]. Ideally, SparseCT can result in a different z-undersampling pattern for *each* gantry angle ϑ. However, in practice, the focal spot at the X-ray source has a finite size and creates penumbra on both sides of each undersampled beam. As a result, each undersampled beam (the beam through one slit of the MSC) irradiates several continuous detector rows. To avoid this overlap between adjacent undersampled beams the distance between two consecutive slits of the MSC needs to be increased. An example of such an undersampling mask, the W4S16 sampling pattern, is depicted in Figure 6.21, where 4 detector-rows are irradiated by each undersampled

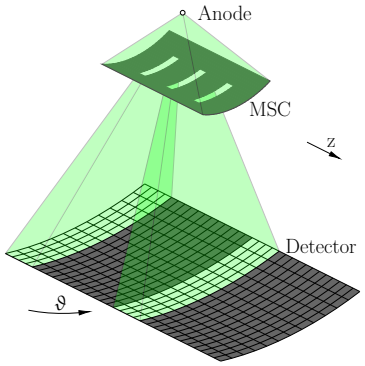


Figure 6.20: Schematic sketch of SparseCT undersampling, where the MSC blocks the majority of the X-ray radiation.

beam and 16 is the distance between two consecutive slits (in detector-row units). The figure also illustrates the temporal evolution of the undersampling pattern with increasing gantry angle ϑ.

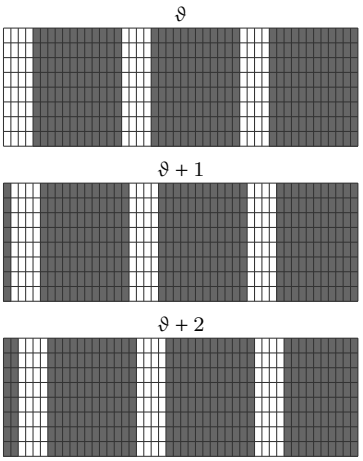


Figure 6.21: Illustration of the W4S16 undersampling mask using a width of 4 detector rows and a stride of 16 rows. Here, the mask moves with collimator speed $v = 1$ detector row per gantry angle, which leads to a different undersampling pattern for each gantry angle.

In SparseCT, image reconstruction is performed by an iterative algorithm using a sparsity-enforcing regularizer on image gradients [16]. The incorporated regularizer is usually a very simple function, such as TV or total generalized variation (TGV), which is not able to fully represent the complex nature of CT images, and might lead to reconstruction artifacts, such as stair-casing, residual streaking artifacts, or image blurring.

To overcome these problems, the new generation of image reconstruction techniques is based on deep learning, where CNNs trained on several datasets are applied in the reconstruction process. Deep learning has been extensively applied to the low-dose CT case for denoising a filtered back-projection (FBP) reconstruction of either low mA data [193–195] or undersampled data [260]. Later, deep learning approaches were incorporated in model-based iterative reconstruction [63, 67, 200, 261], where data consistency is integrated by penalizing the deviation to the acquired sinogram data using a well-known model of the acquisition operator. Hence, the main contribution of deep learning for model-based reconstruction is the improvement of the regularizer.

6.6.1 Model-based SparseCT Reconstruction

Next, we outline the model-based reconstruction of the attenuation volume given the measured sinogram data of a CT scanner using SparseCT undersampling. In detail, we consider the scanned volume $\Omega \subset \mathbb{R}^3$, which is discretized by $n = n_z \cdot n_y \cdot n_x$ voxels. The reconstructed volume is denoted by $x \in \mathbb{R}^n$. Likewise, $z \in \mathbb{R}^m$ with $m = r \cdot a \cdot d_c \cdot d_r$ represents the acquired sinogram data for r rotations of the gantry each measuring a angles, d_c detector channels, and d_r detector rows. Here, we consider a CT scanner with $a = 2304$ angles per gantry rotation and a detector using $d_c = 736$ channels and $d_r = 64$ rows. Each detector element measures a radiation intensity $\iota_i \in \mathbb{R}$, which is determined by the attenuation $\nu : \Omega \to \mathbb{R}$ in the cone Γ_i spanned by the X-ray point source and the rectangular-shaped detector element, i.e.

$$\iota_i = \iota_0 \exp\left(-\int_{\Gamma_i} \nu(s)\mathrm{d}s\right).$$

The X-ray source intensity is denoted by $\iota_0 \in \mathbb{R}$. A transformation into the negative logarithmic domain yields

$$z_i = -\log\left(\frac{\iota_i}{\iota_0}\right) = \int_{\Gamma_i} \nu(s)\mathrm{d}s \approx a_i^T x,$$

where $a_i \in \mathbb{R}^n$ approximates the volumetric integral using lookup tables proposed by [252]. Stacking all post-log sinogram measurements into a vector z yields

$$z = M(Ax + \xi) \tag{6.17}$$

with $A = (a_1, \ldots, a_m)^T \in \mathbb{R}^{m \times n}$. We follow [15, 16] to account for the quantum and electronic noise in the acquisition process and model the noise by a Gaussian distribution $\xi \sim \mathcal{N}(0, W^{-1})$, where W is diagonal with statistical weights that are determined by the flux measurements $W_{ii} = \iota_i$. To retrospectively apply SparseCT undersampling we utilize a binary diagonal mask $M \in \{0, 1\}^{m \times m}$ to the fully-sampled sinogram data z, where the diagonal entries indicate if the corresponding detector element was measured. Using this model, the regularized reconstruction problem is

given by

$$\min_{x\in\mathbb{R}^n} \lambda \mathrm{R}(x) + \tfrac{1}{2}\|M(Ax - z)\|_W^2, \tag{6.18}$$

where $\lambda > 0$ balances the ℓ^2-data consistency term and the regularization term $\mathrm{R} : \mathbb{R}^n \to \mathbb{R}$. Typical choices for the regularizer in medical imaging are TV, TGV, or sparsity-promoting non-convex regularizers on image gradients [45, 262]. All these regularization methods solely quantify local image statistics and are prone to staircasing and blurring artifacts. To overcome these issues, we propose to learn a multi-scale regularizer suitable in particular for CT reconstruction.

6.6.2 Learning a TDV Regularizer for 3D SparseCT

In the above sections, we have rigorously analyzed the TDV regularizer and showed how to efficiently train its parameters θ using early stopping and a mean-field optimal control formulation. Moreover, we showed that the TDV regularizer trained solely for Gaussian denoising can be successfully applied to medical image reconstruction tasks *without* any further training. In what follows, we adopt this TDV regularizer to 3-dimensional CT reconstruction. As introduced above, we consider the TDV regularizer defined by

$$\mathrm{R}(x, \theta) = \sum_{i=1}^{n} r(x, \theta)_i, \tag{6.19}$$

which is the sum of the voxel-wise deep variation $\mathrm{r}(x, \theta) = w^T \mathcal{N}(Kx) \in \mathbb{R}^n$. In detail, $K \in \mathbb{R}^{nm\times n}$ represents a learned 3D convolution kernel with zero-mean constraint. As before, the non-linear function $\mathcal{N} : \mathbb{R}^{nm} \to \mathbb{R}^{nm}$ is a CNN and $w \in \mathbb{R}^m$ is a learned weight vector. However, all convolutions in the residual building blocks of $\mathcal{N}$ operate on 3D volumetric features. To enable an efficient implementation of the regularizer, all 3D convolutions are computed by first applying a 2D convolution in x/y-plane followed by a 1D convolution along the z-axis. The support of the resulting convolution filter is $3 \times 3 \times 3$. We use $m = 16$ feature channels, just a single block, and three scales to reduce the memory footprint of the TDV regularizer.

To estimate the parameters θ of TDV we again consider a sampled discrete optimal control problem. To this end, let $(x_0^i, y^i)_{i=1}^N$ be a collection of N training pairs consisting of an initial degraded volume x_0^i and a corresponding fully-sampled reference y^i. Then, the optimal control training problem is

$$\inf_{T,\theta\in\Theta} \frac{1}{N} \sum_{i=1}^{N} \|x_S^i - y^i\|_\epsilon \tag{6.20}$$

subject to

$$x_{s+1}^i = \frac{1}{1 + \frac{T}{S}} \left(x_s^i + \frac{T}{S} \left(x_0^i - D_1 \mathrm{R}(x_s^i, \theta) \right) \right) \tag{6.21}$$

for $s = 1, \dots, S-1$ and $i = 1, \dots, N$. Here $\|\cdot\|_\epsilon$ is the Huber-norm, $T \in \mathbb{R}^+$ is the learned stopping time, and $S \in \mathbb{N}$ denotes the number of temporal steps. The iterative scheme (6.21) is motivated by a semi-implicit discretization of a gradient flow on the energy $\mathrm{R}(x, \theta) + \frac{1}{2}\|x - x_0^i\|_2^2$, which is equivalent to a proximal gradient method [99].

We use the sinogram data of 10 fully-sampled clinical 3D in-vivo abdominal CT scans of different patients acquired by a Siemens Definition AS scanner using a routine clinical dose. Two of these scans include metal artifacts. We select those two scans for testing, while the remaining scans are used for training. Due to the limited memory

of a single graphics processing unit (GPU), we split the fully-sampled sinogram data of each scan into chunks consisting of 3 full gantry rotations that overlap by 1 rotation, resulting in 133 training samples. We compute the reference volumes y^i by solving the regularized inverse problem (6.18) to account for the noise in the clinical sinogram data. In detail, we choose the commonly used regularizer $\mathrm{R}(x) = \sum_{j=1}^{3n} \frac{1}{2\alpha} \log(1 + \alpha(Dx)_j^2)$ for $\alpha = 1000$ and $\lambda = 0.0125$ using the fully-sampled data (i.e. M is the identity matrix) and an accelerated ordered-subset approach with 8 subsets. Here, D refers to the voxel-wise finite difference operator. The initial reconstructions x_0^i are computed by performing 40 CG steps to solve (6.18) with $\lambda = 0$ for the SparseCT data, where M reflects the W4S16 undersampling with a collimator speed of $v \in \{1, 4\}$ detector rows between two gantry angles.

Given the training samples, we use the Adam optimizer [110] with step size 10^{-4}, $\beta_1 = 0.9$ and $\beta_2 = 0.999$ to estimate the 45 936 parameters of the TDV regularizer and the stopping time T by minimizing (6.20) with $S = 10$. In each of the 50 000 training iterations, 6 patches of size $18 \times 96 \times 96$ are randomly sampled from the training data.

The resulting TDV regularizer with fixed learned parameters θ can be applied to denoise an initial corrupted reconstruction by applying (6.21). Moreover, the variational structure of TDV enables an application in the model-based CT reconstruction setting with balances parameter $\lambda > 0$ as follows:

$$x_R \in \arg\min_{x \in \mathbb{R}^n} \lambda \mathrm{R}(x, \theta) + \tfrac{1}{2} \|M(Ax - z)\|_W^2. \tag{6.22}$$

6.6.3 Numerical Results

The reconstruction of abdominal data with 4-fold retrospective SparseCT undersampling is presented in Figure 6.22 and Figure 6.23. The first columns in both figures show the initial non-regularized CG reconstruction with undersampling artifacts, the second columns depict the denoising results of the CG initialization utilizing the TDV regularizer and (6.21), the third column displays the TDV reconstruction results using (6.22), and the last columns show fully-sampled reference images. We present two cases of SparseCT undersampling that correspond to two different collimator speeds v of the MSC:

- $v = 4$ detector-rows per gantry angle (higher undersampling performance, Figure 6.22) and
- $v = 1$ detector-rows per gantry angle (lower undersampling performance, Figure 6.23).

In both cases, TDV reconstruction outperforms TDV denoising, and presents images that are closer to the reference images — qualitatively and in terms of PSNR and SSIM. While the TDV denoising results are degraded by blurring artifacts, the incorporation of data consistency in TDV reconstruction enables to reinsert high-resolution features from the raw data, which results in significant improvements of the image quality. As expected, SparseCT with a faster collimator speed $v = 4$ outperforms $v = 1$ due to its superior undersampling performance. However, the difference is lower in the case of TDV reconstruction with respect to TDV denoising, which shows the ability of the latter to adapt to situations with data of lower acquisition quality.

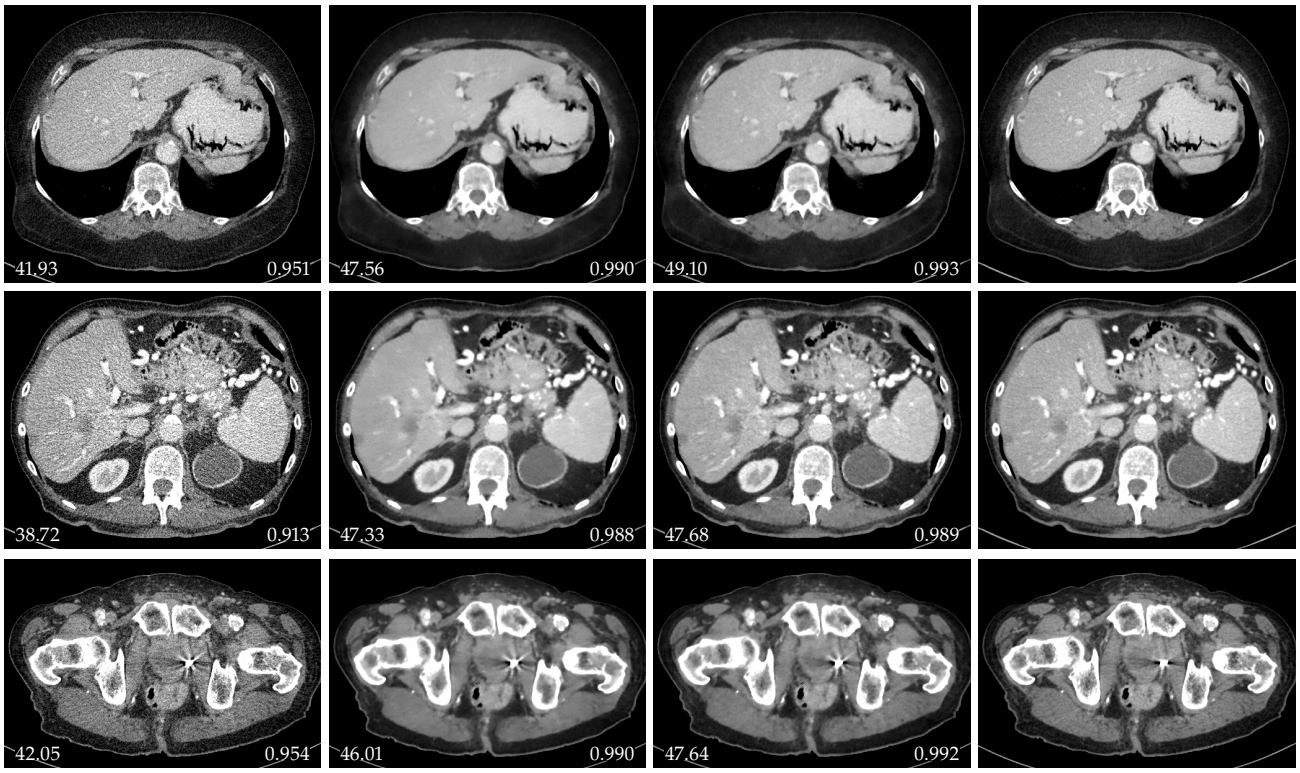


Figure 6.22: Reconstruction of 4-fold undersampled SparseCT sinogram data with collimator speed $v = 4$ detector-rows per gantry angle. From left to right: unregularized CG initialization, TDV denoising, TDV reconstruction, fully-sampled reference. Images are displayed in the range 40 ± 200HU. The PSNR scores w.r.t. to the reference y are depicted on the lower left corner and the SSIM score in the lower right corner.

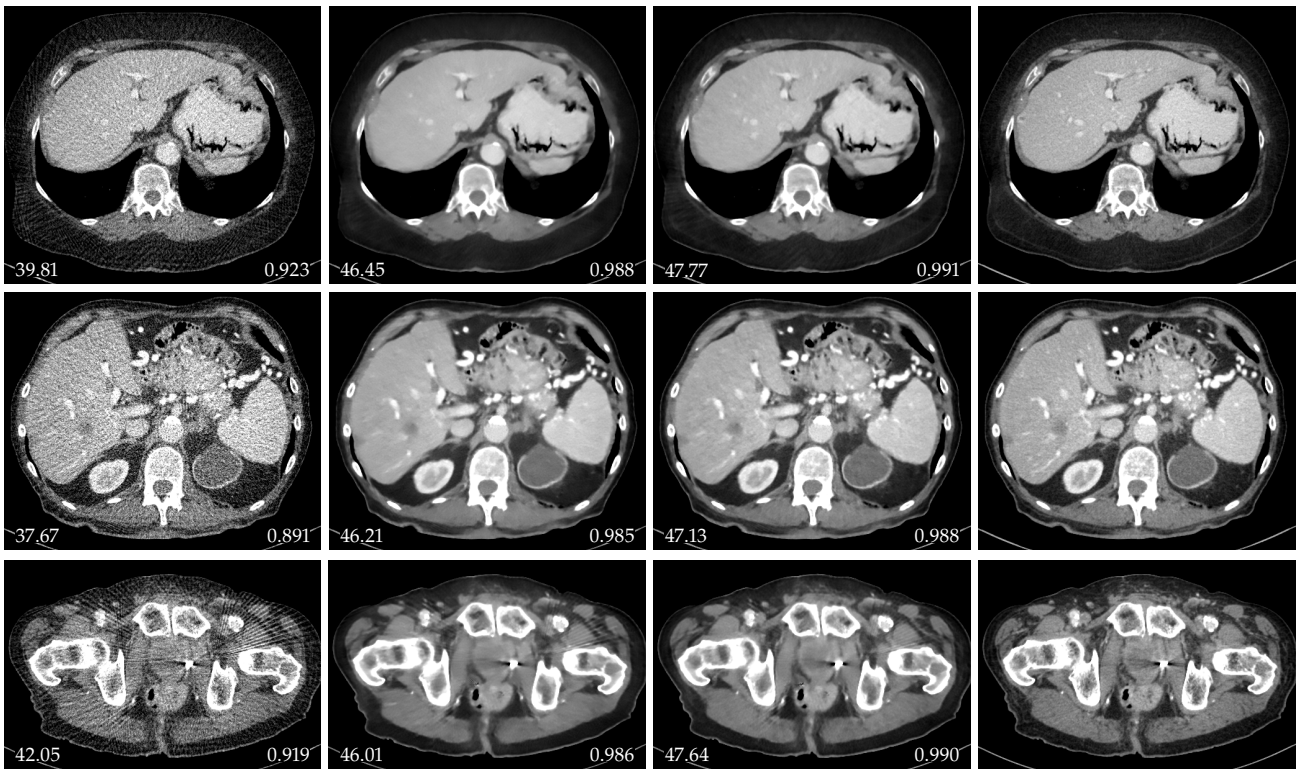


Figure 6.23: Reconstruction of 4-fold undersampled SparseCT sinogram data with collimator speed $v = 1$ detector-rows per gantry angle. From left to right: unregularized CG initialization, TDV denoising, TDV reconstruction, fully-sampled reference. Images are displayed in the range 40 ± 200HU. The PNSR scores w.r.t. to the reference y are depicted on the lower left corner and the SSIM score in the lower right corner.

6.7 Conclusion

The proposed total deep variation regularizer is motivated by established deep network architectures. Moreover, the inherent variational structure of our approach presented in this chapter enables a rigorous mathematical understanding encompassing an optimality condition for optimal stopping, and a nonlinear eigenfunction analysis. We have derived theoretical upper bounds for the stability analysis, which led to relatively tight bounds in the numerical experiments. For image denoising and single image super-resolution, our model generates state-of-the-art results with an impressively low number of trainable parameters. To underline the versatility of TDV for generic linear inverse problems, we successfully demonstrated their applicability for the challenging CT and MRI reconstruction tasks without requiring any additional training. In addition, we have conducted adversarial attacks and an empirical worst-case generalization error analysis to demonstrate the robustness of our approach.

We extended the TDV regularizer to 3-dimensional helical cone-beam CT reconstruction and proposed an efficient training approach that avoids the memory and time consuming linear data consistency operators in the training process. The resulting TDV regularizer can be used in a variational reconstruction setting, which leads to superior results compared to using the same TDV regularizer to denoising an initially corrupted reconstruction. To sum up, using the TDV regularizer for reconstruction is a promising alternative to classical sparsity enforcing regularizers and/or deep learning reconstruction schemes due to its variational structure and performance, and can lead to high CT dose reduction with little impact on image quality.

7 Conclusion and Outlook

As outlined in the introduction, numerous tasks in computer vision and medical imaging can be framed as inverse problems. To account for the ill-posed nature of inverse problems and to derive robust estimates of their solutions, we have to combine domain knowledge of the acquisition process with prior knowledge about properties of the solution. For this task, variational methods provide a very flexible framework. They amount to minimizing an energy functional composed of a data fidelity term and a regularizer. While the data fidelity term is typically used to incorporate knowledge about the acquisition process, the regularizer characterizes properties of the solution itself. In fact, the statistical interpretation of inverse problems highlights that the regularizer represents the negative log-likelihood of the prior distribution of desired solutions. However, modeling this prior distribution by hand is challenging. While the statistics of image gradients can be effectively modeled by Laplace or Student-t distributions, higher-order statistics of local pixel neighborhoods such as 2×2 image patches are much more complicated to capture — as we saw in Chapter 4. This motivates the application of machine learning to extract higher-order prior information from data to improve the estimation of solutions of inverse problems.

In this thesis, we have presented different approaches that combine effective data-driven machine learning models with the rich theoretical understanding of variational methods to solve inverse problems. We first introduced variational networks (VNs) that establish links between variational methods and deep learning. A VN stacks several parametric incremental proximal gradient steps, which can be adapted to learn proximal gradient schemes, incremental proximal gradient schemes, or trainable nonlinear reaction diffusion (TNRD) models. In addition, VNs are equivalent to residual neural networks (ResNets) if a gradient structure within the residual update steps is assumed. The versatility of VNs enabled a detailed numerical comparison of these schemes and allowed us to investigate the limitations of convexity in the context of proximal gradient reconstruction schemes. Our findings for the inverse problems of image denoising and deblurring show that schemes associated with a convex energy yielded inferior results than their non-convex counterparts. Moreover, VNs representing incremental schemes only require a few steps to yield reasonable results even for the challenging tasks of non-blind deblurring and 3-dimensional undersampled computed tomography (CT) reconstruction.

To analyze this phenomenon, we formulated VNs as an early stopped time-continuous gradient flow of an energy composed of a quadratic data fidelity term and a parametric regularizer. We advocated the fields of experts (FoE) regularizer and introduced the total deep variation (TDV) regularizer to overcome the limited complexity of the FoE regularizer, which is essentially determined by a single convolution layer in conjunction with a parametric nonlinear potential function. In contrast, the TDV regularizer is a convolutional neural network (CNN) that extracts image features on multiple scales and processes them in multiple blocks to determine a local energy for each pixel within an image. The parameters of the regularizer as well as the optimal stopping time of the gradient flow were learned from data by a mean-field optimal control problem, which enabled a rigorous mathematical analysis. We proved the existence of solutions of this mean-field optimal control problem in the time-continuous and time-discrete setting utilizing a semi-implicit time-discretization. Furthermore, a first-order optimality condition for the optimal stopping time as well as stochastic upper bounds regarding the stability of input and parameter variations of the proposed approach were derived. We numerically verified the first-order condition and showed that the proposed stability bounds are fairly tight. Our

proposed approach featuring the TDV regularizer yields state-of-the-art numerical results for image denoising as well as single image super-resolution (SISR). The broad applicability of the TDV regularizer was demonstrated by applying it to accelerated magnetic resonance imaging (MRI) and 2-dimensional angular-undersampled CT reconstruction. In addition, we presented an efficient training strategy to learn a tailored TDV regularizer for 3-dimensional SparseCT reconstruction.

Due to the versatility of the proposed approaches, there are many possible future research directions. First of all, extending our approach to learn the optimal stopping time of non-autonomous gradient flows is interesting. This would enable the analysis of gradient flows whose parameters are allowed to change over time. However, a major issue related to this extension originates from the continuation of the stopping time beyond its optimal point because it is not clear how to extrapolate the dynamic model parameters.

Further possible research directions involve the development of parametric energies. Nonlocal methods [72, 73] have proven to be particularly successful in the presence of strong noise. Therefore, a promising research direction is to develop parametric regularizers that determine a pixel-wise energy based on nonlocal information extracted from a larger pixel neighborhood. An alternative approach is to determine the pixel-wise energy of a parametric regularizer over a multi-scale representation of an image, in analogy to the multi-scale correlation volume advocated in [125]. In this thesis, we mainly assumed that the measurement noise is Gaussian and utilized a quadratic data fidelity term. However, in many real-world applications we do not know the exact distribution of the noise. Consequently, further development of the data fidelity term is another promising research direction.

Other open research directions address the parameter identification of the proposed TDV regularizer. In this thesis, we have presented a discriminative learning approach to determine the parameters of the TDV regularizer and we have demonstrated that the resulting generic regularizers can be applied to different inverse problems. However, the learned TDV regularizers emphasize task-specific properties since they are trained in a discriminative way, as indicated by the nonlinear eigenfunction analysis. It is not clear whether these effects can be avoided by advocating a generative learning approach. Additionally, it would be interesting to further analyze the sensitivity of the solution to different initial images and compare bilevel optimization with the proposed training scheme in this regard.

Finally, a promising research direction is to utilize the statistical information encoded in the associated posterior probability. So far, we have only computed the maximum a-posteriori (MAP) estimator. Indeed, the variational structure of the proposed approach allows us to draw samples from the posterior distribution [263]. Using these samples, we could, for instance, estimate the expectation and variance of the posterior to account for the uncertainty in the restoration process, as advocated in [264].

Appendix

List of Publications A

2017

Erich Kobler, Teresa Klatzer, Kerstin Hammernik, and Thomas Pock. "Variational Networks: Connecting Variational Methods and Deep Learning". In: *German Conference on Pattern Recognition*. 2017

Teresa Klatzer, Daniel Soukup, Erich Kobler, Kerstin Hammernik, and Thomas Pock. "Trainable regularization for multi-frame superresolution". In: *German Conference on Pattern Recognition*. 2017

2018

Alexander Effland, Michael Hölzel, Teresa Klatzer, Erich Kobler, Jennifer Landsberg, Leonie Neuhäuser, Thomas Pock, and Martin Rumpf. "Variational Networks for Joint Image Reconstruction and Classification of Tumor Immune Cell Interactions in Melanoma Tissue Sections". In: *Bildverarbeitung für die Medizin*. 2018

Erich Kobler, Matthew J. Muckley, Baiyu Chen, Florain Knoll, Kerstin Hammernik, Thomas Pock, Daniel K. Sodickson, and Ricardo Otazo. "Variational network learning for low-dose CT". In: *Proceedings of the* 5^{th} *CT Meeting*. 2018

Kerstin Hammernik, Erich Kobler, Thomas Pock, Michael P. Recht, Daniel K. Sodickson, and Florian Knoll. "Variational adversarial networks for accelerated MR image reconstruction". In: *Proceedings of the International Society of Magnetic Resonance in Medicine*. 2018

Kerstin Hammernik, Teresa Klatzer, Erich Kobler, Michael P. Recht, Daniel K. Sodickson, Thomas Pock, and Florian Knoll. "Learning a variational network for reconstruction of accelerated MRI data". In: *Magn. Reson. Med.* 79.6 (2018)

Florian Knoll, Kerstin Hammernik, Erich Kobler, Thomas Pock, Daniel K. Sodickson, and Michael P. Recht. "Analysis of the influence of deviations between training and test data in learned image reconstruction". In: *ISMRM Workshop on Machine Learning*. 2018

2019

Florian Knoll, Kerstin Hammernik, Erich Kobler, Thomas Pock, Michael P. Recht, and Daniel K. Sodickson. "Assessment of the generalization of learned image reconstruction and the potential for transfer learning". In: *Magn. Reson. Med.* 81.1 (2019)

Alexander Effland, Erich Kobler, Anne Brandenburg, Teresa Klatzer, Leonie Neuhäuser, Michael Hölzel, Jennifer Landsberg, Thomas Pock, and Martin Rumpf. "Joint reconstruction and classification of tumor cells and cell interactions in melanoma tissue sections with synthesized training data". In: *Int. J. Comput. Assist. Radiol. Surg.* 14.4 (2019)

Alexander Effland, Erich Kobler, Thomas Pock, and Martin Rumpf. "Time discrete geodesics in deep feature spaces for image morphing". In: *International Conference on Scale Space and Variational Methods in Computer Vision*. 2019

Baiyu Chen, Erich Kobler, Matthew J. Muckley, Aaron D. Sodickson, Thomas O'Donnell, Thomas Flohr, Bernhard Schmidt, Daniel K. Sodickson, and Ricardo Otazo. "SparseCT: System concept and design of multislit collimators". In: *Med. Phys.* 46.6 (2019)

Alexander Effland, Erich Kobler, Karl Kunisch, and Thomas Pock. "Variational Networks: An Optimal Control Approach to Early Stopping Variational Methods for Image Restoration". In: *J. Math. Imaging Vision* 62.3 (2020)

2020

Alexander Effland, Erich Kobler, Thomas Pock, Marko Rajković, and Martin Rumpf. "Image Morphing in Deep Feature Spaces: Theory and Applications". In: *J. Math. Imaging Vision* 62.3 (2020)

Erich Kobler, Alexander Effland, Karl Kunisch, and Thomas Pock. "Total Deep Variation for Linear Inverse Problems". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2020

Martin Zach and Erich Kobler. "Real-World Video Restoration using Noise-2-Noise". In: *Proceedings of the Joint Austrian Computer Vision and Robotics Workshop*. 2020

Elena A. Kaye, Emily A. Aherne, Cihan Duzgol, Ida Häggström, Erich Kobler, Yousef Mazaheri, Maggie M. Fung, Zhigang Zhang, Ricardo Otazo, Herbert A. Vargas, and Oguz Akin. "Accelerating Prostate Diffusion Weighted MRI using Guided Denoising Convolutional Neural Network: Retrospective Feasibility Study". In: *Radiology: Artificial Intelligence* 2.5 (2020)

Erich Kobler, Baiyu Chen, Alexander Effland, Thomas Pock, Daniel K. Sodickson, and Ricardo Otazo. "Total Deep Variation for SparseCT Reconstruction". In: *Proceedings of the 6^{th} CT Meeting*. 2020

Erich Kobler, Alexander Effland, Karl Kunisch, and Thomas Pock. "Total Deep Variation: A Stable Regularizer for Inverse Problems". In: *IEEE Trans. Pattern Anal. Mach. Intell.* (2020). (submitted)

Bibliography

[1] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. "Nonlinear total variation based noise removal algorithms". In: *Phys. D* 60.1-4 (1992), pp. 259–268 (cited on pages 1, 4, 79, 90, 91, 99, 109, 111).

[2] Anat Levin, Yair Weiss, Fredo Durand, and William T. Freeman. "Understanding and Evaluating Blind Deconvolution Algorithms". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 1964–1971 (cited on pages 1, 90, 97).

[3] David Mumford and Jayant Shah. "Optimal Approximations by Piecewise Smooth Functions and Associated Variational Problems". In: *Communications on Pure and Applied Mathematics* 42.5 (1989), pp. 577–685 (cited on pages 1, 90).

[4] Berthold K.P. Horn and Brian G. Schunck. "Determining optical flow". In: *Techniques and Applications of Image Understanding*. 1981, pp. 319–331 (cited on pages 1, 4, 90).

[5] Jeffrey A. Fessler. "Penalized weighted least-squares image reconstruction for positron emission tomography". In: *IEEE Trans. Med. Imaging* 13.2 (1994), pp. 290–300 (cited on pages 1, 4).

[6] Stuart Gaman and D. McClure. "Bayesian image analysis: An application to single photon emission tomography". In: *Amer. Statist. Assoc* (1985), pp. 12–18 (cited on pages 1, 5, 81).

[7] Florian Knoll, Kristian Bredies, Thomas Pock, and Rudolf Stollberger. "Second order total generalized variation (TGV) for MRI". In: *Magn. Reson. Med.* 65.2 (2011), pp. 480–491 (cited on page 1).

[8] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521 (2015), pp. 436–444 (cited on page 1).

[9] Jacques Hadamard. *Lectures on Cauchy's problem in linear partial differential equations*. Yale University Press, New Haven, CT, 1923 (cited on pages 1, 4, 77).

[10] Antonin Chambolle and Thomas Pock. "An introduction to continuous optimization for imaging". In: *Acta Numer.* 25 (2016), pp. 161–319 (cited on pages 2, 4, 48, 111, 126, 127).

[11] Eirikur Agustsson and Radu Timofte. "NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study". In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2017 (cited on pages 2, 134).

[12] Cynthia H. McCollough, Adam C. Bartley, Rickey E. Carter, Baiyu Chen, Tammy A. Drees, Phillip Edwards, David R. Holmes III, Alice E. Huang, Farhana Khan, Shuai Leng, Kyle L. McMillan, Gregory J. Michalak, Kristina M. Nunez, Lifeng Yu, and Joel G. Fletcher. "Low-dose CT for the detection and classification of metastatic liver lesions: Results of the 2016 Low Dose CT Grand Challenge". In: *Med. Phys.* 44.10 (2017), pp. 339–352 (cited on pages 2, 126).

[13] Jure Zbontar, Florian Knoll, Anuroop Sriram, Matthew J. Muckley, Mary Bruno, Aaron Defazio, Marc Parente, Krzysztof J. Geras, Joe Katsnelson, Hersh Chandarana, Zizhao Zhang, Michal Drozdzal, Adriana Romero, Michael Rabbat, Pascal Vincent, James Pinkerton, Duo Wang, Nafissa Yakubova, Erich Owens, Lawrence C. Zitnick, Michael P. Recht, Daniel K. Sodickson, and Yvonne W. Lui. "fastMRI: An Open Dataset and Benchmarks for Accelerated MRI". In: *arXiv* 1811.08839 (2018) (cited on pages 2, 6, 50).

[14] Alessandro Foi. "Clipped noisy images: Heteroskedastic modeling and practical denoising". In: *Signal Processing* 89.12 (2009), pp. 2609–2629 (cited on page 2).

[15] Jean-Baptiste Thibault, Ken D Sauer, Charles A Bouman, and Jiang Hsieh. "A three-dimensional statistical approach to improved image quality for multislice helical CT". In: *Med. Phys.* 34.11 (2007), pp. 4526–4544 (cited on pages 3, 139).

[16] Matthew J. Muckley, Baiyu Chen, Thomas Vahle, Thomas O'Donnell, Florian Knoll, Aaron D. Sodickson, Daniel K. Sodickson, and Ricardo Otazo. "Image reconstruction for interrupted-beam x-ray CT on diagnostic clinical scanners". In: *Phys. Med. Biol.* 64 (2019), p. 155007 (cited on pages 3, 139).

[17] Donald W. McRobbie, Elizabeth A. Moore, Martin J. Graves, and Martin R. Prince. *MRI from Picture to Proton*. Cambridge university press, 2017 (cited on page 3).

[18] Emmanuel J. Candès, Justin Romberg, and Terence Tao. "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information". In: *IEEE Trans. on Information Theory* 52.2 (2006), pp. 489–509 (cited on pages 3, 102).

[19] Daniel K. Sodickson and Warren J. Manning. "Simultaneous acquisition of spatial harmonics (SMASH): fast imaging with radiofrequency coil arrays". In: *Magn. Reson. Med.* 38.4 (1997), pp. 591–603 (cited on page 3).

[20] Klaas P. Pruessmann, Markus Weiger, Markus B. Scheidegger, and Peter Boesiger. "SENSE: sensitivity encoding for fast MRI". In: *Magn. Reson. Med.* 42.5 (1999), pp. 952–962 (cited on page 3).

[21] Frank Natterer and Frank Wubbeling. "A propagation-backpropagation method for ultrasound tomography". In: *Inverse problems* 11.6 (1995), p. 1225 (cited on page 4).

[22] Andrey N. Tikhonov. "On the solution of ill-posed problems and the method of regularization". In: *Dokl. Akad. Nauk SSSR* 151 (1963), pp. 501–504 (cited on pages 4, 77).

[23] David L. Phillips. "A technique for the numerical solution of certain integral equations of the first kind". In: *J. Assoc. Comput. Mach.* 9 (1962), pp. 84–97 (cited on pages 4, 78).

[24] Martin Benning and Martin Burger. "Modern regularization methods for inverse problems". In: *Acta Numer.* 27 (2018), pp. 1–111 (cited on page 4).

[25] Barbara Kaltenbacher, Andreas Neubauer, and Otmar Scherzer. *Iterative regularization methods for nonlinear ill-posed problems*. Vol. 6. Radon Series on Computational and Applied Mathematics. Walter de Gruyter GmbH & Co. KG, Berlin, 2008 (cited on pages 4, 110).

[26] Otmar Scherzer, Markus Grasmair, Harald Grossauer, Markus Haltmeier, and Frank Lenzen. *Variational methods in imaging*. Vol. 167. Applied Mathematical Sciences. Springer, New York, 2009 (cited on pages 4, 10).

[27] Antonin. Chambolle and Pierre-Louis Lions. "Image recovery via total variation minimization and related problems". In: *Numer. Math.* 76.2 (1997), pp. 167–188 (cited on pages 4, 79, 80, 91, 109).

[28] Joachim Weickert. *Anisotropic diffusion in image processing*. European Consortium for Mathematics in Industry. B. G. Teubner, Stuttgart, 1998 (cited on page 4).

[29] Kristian Bredies, Karl Kunisch, and Thomas Pock. "Total generalized variation". In: *SIAM J. Imaging Sci.* 3.3 (2010), pp. 492–526 (cited on pages 4, 80, 91, 109).

[30] Matthias J. Ehrhardt and Marta M. Betcke. "Multicontrast MRI reconstruction with structure-guided total variation". In: *SIAM J. Imaging Sci.* 9.3 (2016), pp. 1084–1106 (cited on page 4).

[31] Andrew M. Stuart. "Inverse problems: a Bayesian perspective". In: *Acta Numer.* 19 (2010), pp. 451–559 (cited on pages 4, 69).

[32] Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006 (cited on pages 5, 52, 55–57, 59, 61, 64, 69, 78).

[33] Stuart Geman and Donald Geman. "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 6 (1984), pp. 721–741 (cited on pages 5, 81, 82).

[34] Kevin P. Murphy. *Machine learning: a probabilistic perspective*. Cambridge, MA: MIT Press, 2012 (cited on pages 5, 52, 53, 55–59, 69, 78, 81).

[35] David J. Field. "Relations between the statistics of natural images and the response properties of cortical cells". In: *J. Opt Soc. Am. A* 4.12 (1987), pp. 2379–2394 (cited on pages 5, 73, 77).

[36] Daniel L. Ruderman and William Bialek. "Statistics of natural images: Scaling in the woods". In: *Advances in Neural Information Processing Systems*. 1994, pp. 551–558 (cited on pages 5, 73, 77).

[37] Peter J. Huber. *Robust statistics*. John Wiley & Sons, Inc., New York, 1981 (cited on pages 5, 46, 81).

[38] Peter J. Green. "Bayesian reconstructions from emission tomography data using a modified EM algorithm". In: *IEEE Trans. on Med. Imaging* 9.1 (1990), pp. 84–93 (cited on pages 5, 81).

[39] Pierre Charbonnier, Laure Blanc-Féraud, Gilles Aubert, and Michel Barlaud. "Deterministic edge-preserving regularization in computed imaging". In: *IEEE Trans. on Image Processing* 6.2 (1997), pp. 298–311 (cited on pages 5, 79, 81, 84).

[40] D. F. Andrews, P. J. Bickel, F. R. Hampel, P. J. Huber, W. H. Rogers, and J. W. Tukey. *Robust estimates of location: Survey and advances*. Princeton University Press, Princeton, N.J., 1972 (cited on pages 5, 81).

[41] Andrew Blake and Andrew Zisserman. *Visual reconstruction*. MIT Press, Cambridge, MA, 1987 (cited on pages 5, 81, 82, 91).

[42] Tom Hebert and Richard Leahy. "A generalized EM algorithm for 3-D Bayesian reconstruction from Poisson data using Gibbs priors". In: *IEEE Trans. on Med. Imaging* 8.2 (1989), pp. 194–202 (cited on pages 5, 81).

[43] Jinggang Huang and David Mumford. "Statistics of natural images and models". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 1999, pp. 541–547 (cited on pages 5, 73, 81, 82, 86).

[44] Eero P. Simoncelli. "Modeling the joint statistics of images in the wavelet domain". In: *Proc. SPIE 44th Annual Meeting*. Vol. 3813. 1999, pp. 188–195 (cited on pages 5, 73, 82).

[45] Song Chun Zhu and David Mumford. "Prior learning and Gibbs reaction-diffusion". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 19.11 (1997), pp. 1236–1250 (cited on pages 5, 82, 91, 140).

[46] Song Chun Zhu, Yingnian Wu, and David Mumford. "Filters, Random Fields and Maximum Entropy (FRAME): Towards a Unified Theory for Texture Modeling". In: *Int. J. Comput. Vision* 27.2 (1998), pp. 107–126 (cited on pages 5, 82, 109).

[47] Geoffrey E Hinton. "Products of Experts". In: *International Conference on Artificial Neural Networks*. 1999, pp. 1–6 (cited on pages 5, 83).

[48] Geoffrey E Hinton. "Training products of experts by minimizing contrastive divergence". In: *Neural Comput.* 14.8 (2002), pp. 1771–1800 (cited on pages 5, 83, 87).

[49] Max Welling, Simon Osindero, and Geoffrey E. Hinton. "Learning sparse topographic representations with products of student-t distributions". In: *Advances in Neural Information Processing Systems*. 2003, pp. 1383–1390 (cited on pages 5, 83).

[50] Stefan Roth and Michael J. Black. "Fields of experts: A framework for learning image priors". In: *IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 2. 2005, pp. 860–867 (cited on page 5).

[51] Stefan Roth and Michael J. Black. "Fields of Experts". In: *Int. J. Comput. Vis.* 82.2 (2009), pp. 205–229 (cited on pages 5, 82–84, 91, 103, 109, 113, 114).

[52] Kegan G. G. Samuel and Marshall F. Tappen. "Learning optimized MAP estimates in continuously-valued MRF models". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 477–484 (cited on pages 5, 82, 87, 88, 109).

[53] Yunjin Chen, Rene Ranftl, and Thomas Pock. "Insights into analysis operator learning: From patch-based sparse models to higher order MRFs". In: *IEEE Trans. on Image Processing* 23.3 (2014), pp. 1060–1072 (cited on pages 5, 82, 87, 88, 91).

[54] Justin Domke. "Generic Methods for Optimization-Based Modeling". In: *International Conference on Artificial Intelligence and Statistics*. 2012, pp. 318–326 (cited on pages 5, 88, 91, 109).

[55] Yunjin Chen and Thomas Pock. "Trainable Nonlinear Reaction Diffusion: A Flexible Framework for Fast and Effective Image Restoration". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 39.6 (2017), pp. 1256–1272 (cited on pages 6–8, 84, 91, 92, 95, 99, 103, 109, 125).

[56] Erich Kobler, Teresa Klatzer, Kerstin Hammernik, and Thomas Pock. "Variational Networks: Connecting Variational Methods and Deep Learning". In: *German Conference on Pattern Recognition*. 2017, pp. 281–293 (cited on pages 6, 7, 82, 90, 95, 103, 104, 109, 147).

[57] Alexander Effland, Erich Kobler, Karl Kunisch, and Thomas Pock. "Variational Networks: An Optimal Control Approach to Early Stopping Variational Methods for Image Restoration". In: *J. Math. Imaging Vision* 62.3 (2020), pp. 396–416 (cited on pages 6, 7, 82, 108, 109, 148).

[58] Weinan E, Jiequn Han, and Qianxiao Li. "A mean-field optimal control formulation of deep learning". In: *Res. Math. Sci.* 6.1 (2019), Paper No. 10, 41 (cited on pages 6, 66, 110, 112, 113).

[59] Erich Kobler, Alexander Effland, Karl Kunisch, and Thomas Pock. "Total Deep Variation for Linear Inverse Problems". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2020 (cited on pages 6, 7, 85, 108, 113, 148).

[60] Erich Kobler, Alexander Effland, Karl Kunisch, and Thomas Pock. "Total Deep Variation: A Stable Regularizer for Inverse Problems". In: *IEEE Trans. Pattern Anal. Mach. Intell.* (2020). (submitted) (cited on pages 6, 7, 82, 85, 108, 113, 148).

[61] Sebastian Lunz, Ozan Öktem, and Carola-Bibiane Schönlieb. "Adversarial regularizers in inverse problems". In: *Advances in Neural Information Processing Systems*. 2018, pp. 8507–8516 (cited on pages 6, 82, 87, 109).

[62] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein generative adversarial networks". In: *International Conference on Machine Learning*. 2017, pp. 214–223 (cited on page 6).

[63] Housen Li, Johannes Schwab, Stephan Antholzer, and Markus Haltmeier. "NETT: Solving Inverse Problems with Deep Neural Networks". In: *Inverse Problems* (2020) (cited on pages 6, 82, 109, 139).

[64] Karol Gregor and Yann LeCun. "Learning fast approximations of sparse coding". In: *International Conference on Machine Learning*. 2010, pp. 399–406 (cited on pages 6, 91).

[65] Christoph Vogel and Thomas Pock. "A primal dual network for low-level vision problems". In: *German Conference on Pattern Recognition*. 2017, pp. 189–202 (cited on page 6).

[66] Antonin Chambolle and Thomas Pock. "A first-order primal-dual algorithm for convex problems with applications to imaging". In: *J. Math. Imaging Vision* 40.1 (2011), pp. 120–145 (cited on pages 6, 49, 80).

[67] Jonas Adler and Ozan Öktem. "Learned primal-dual reconstruction". In: *IEEE Trans. Med. Imaging* 37.6 (2018), pp. 1322–1332 (cited on pages 6, 55, 139).

[68] Singanallur V. Venkatakrishnan, Charles A. Bouman, and Brendt Wohlberg. "Plug-and-Play priors for model based reconstruction". In: *IEEE Global Conference on Signal and Information Processing*. 2013, pp. 945–948 (cited on page 6).

[69] Yaniv Romano, Michael Elad, and Peyman Milanfar. "The little engine that could: regularization by denoising (RED)". In: *SIAM J. Imaging Sci.* 10.4 (2017), pp. 1804–1844 (cited on page 6).

[70] Jonathan Eckstein and Dimitri P. Bertsekas. "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators". In: *Math. Programming* 55.3, Ser. A (1992), pp. 293–318 (cited on page 6).

[71] Stephen Boyd, Neal Parikh, and Eric Chu. "Distributed optimization and statistical learning via the alternating direction method of multipliers". In: *Found. Trends Mach. Learn.* 3.1 (2011), pp. 1–122 (cited on page 6).

[72] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. "A non-local algorithm for image denoising". In: *IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 2. 2005, pp. 60–65 (cited on pages 6, 77, 145).

[73] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. "Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering". In: *IEEE Trans. on Image Processing* 16.8 (2007), pp. 2080–2095 (cited on pages 6, 99, 125, 145).

[74] Tim Meinhardt, Michael Möller, Caner Hazirbas, and Daniel Cremers. "Learning proximal operators: Using denoising networks for regularizing inverse imaging problems". In: *IEEE International Conference on Computer Vision*. 2017, pp. 1781–1790 (cited on page 6).

[75] J. H. Rick Chang, Chun-Liang Li, Barnábas Póczos, B. V. K. Vijaya Kumar, and Aswin C. Sankaranarayanan. "One Network to Solve Them All — Solving Linear Inverse Problems Using Deep Projection Models". In: *IEEE International Conference on Computer Vision*. 2017, pp. 5888–5897 (cited on page 6).

[76] Simon Arridge, Peter Maaß, Ozan Öktem, and Carola-Bibiane Schönlieb. "Solving inverse problems using data-driven models". In: *Acta Numer.* 28 (2019), pp. 1–174 (cited on page 6).

[77] Florian Knoll, Kerstin Hammernik, Chi Zhang, Steen Moeller, Thomas Pock, Daniel K Sodickson, and Mehmet Akcakaya. "Deep-learning methods for parallel magnetic resonance imaging reconstruction: A survey of the current approaches, trends, and issues". In: *IEEE Signal Process. Mag.* 37.1 (2020), pp. 128–140 (cited on page 6).

[78] Johannes Leuschner, Maximilian Schmidt, Daniel Otero Baguer, and Peter Maaß. "The lodopab-ct dataset: A benchmark dataset for low-dose ct reconstruction methods". In: *arXiv* 1910.01113 (2019) (cited on pages 6, 50).

[79] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. "Tensorflow: A system for large-scale machine learning". In: *12th USENIX Symposium on Operating Systems Design and Implementation OSDI*. 2016, pp. 265–283 (cited on pages 7, 62).

[80] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. "Pytorch: An imperative style, high-performance deep learning library". In: *Advances in Neural Information Processing Systems*. 2019, pp. 8026–8037 (cited on pages 7, 62).

[81] Dimitri P. Bertsekas. "Incremental gradient, subgradient, and proximal methods for convex optimization: A survey". In: *Optimization for Machine Learning* 2010.1-38 (2011), p. 3 (cited on pages 7, 50, 51, 91–93).

[82] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification". In: *International Conference on Computer Vision*. 2015, pp. 1026–1034 (cited on pages 7, 63).

[83] Thomas Kösters, Florian Knoll, Aaron D. Sodickson, Daniel K. Sodickson, and Ricardo Otazo. "SparseCT: interrupted-beam acquisition and sparse reconstruction for radiation dose reduction". In: *Medical Imaging 2017: Physics of Medical Imaging*. SPIE, 2017, pp. 174–180 (cited on pages 7, 102, 138).

[84] Amir Beck. *First-order methods in optimization*. Vol. 25. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA; Mathematical Optimization Society, Philadelphia, PA, 2017 (cited on pages 10, 17, 18, 36, 43–46).

[85] Hans W. Alt. *Linear functional analysis*. Universitext. Springer-Verlag London, Ltd., London, 2016 (cited on pages 10, 13, 14, 25).

[86] Dirk Werner. *Funktionalanalysis*. Springer-Verlag, Berlin, 2000 (cited on page 10).

[87] Achim Klenke. *Probability theory*. Second. Universitext. Springer, London, 2014 (cited on pages 10, 15, 20–25, 28–31, 86).

[88] Gerald Teschl. *Ordinary differential equations and dynamical systems*. Vol. 140. Graduate Studies in Mathematics. American Mathematical Society, Providence, RI, 2012 (cited on pages 32, 34, 35, 115, 116).

[89] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, 2004 (cited on pages 36, 43).

[90] Amir Beck. *Introduction to nonlinear optimization*. Vol. 19. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA; Mathematical Optimization Society, Philadelphia, PA, 2014 (cited on pages 36–43, 46, 47, 95).

[91] Yurii Nesterov. *Introductory lectures on convex optimization*. Kluwer Academic Publishers, Boston, MA, 2004 (cited on pages 36, 47, 48).

[92] Yurii Nesterov. *Lectures on convex optimization*. Springer, Cham, 2018 (cited on pages 36, 40, 41, 43, 47).

[93] Martin Barner and Friedrich Flohr. *Analysis II*. De Gruyter, Berlin, 1996 (cited on page 37).

[94] A. S. Nemirovsky and D. B. Yudin. *Problem complexity and method efficiency in optimization*. A Wiley-Interscience Publication. John Wiley & Sons, Inc., New York, 1983 (cited on page 47).

[95] Yuri E. Nesterov. "Smooth minimization of non-smooth functions". In: *Math. Program.* 103.1 (2005), pp. 127–152 (cited on pages 47, 49).

[96] Coralia Cartis, Nick I. M. Gould, and Philippe L. Toint. "On the complexity of steepest descent, Newton's and regularized Newton's methods for nonconvex unconstrained optimization problems". In: *SIAM J. Optim.* 20.6 (2010), pp. 2833–2852 (cited on page 47).

[97] Saeed Ghadimi and Guanghui Lan. "Accelerated gradient methods for nonconvex nonlinear and stochastic programming". In: *Math. Program.* 156.1-2 (2016), pp. 59–99 (cited on pages 47–49).

[98] Yuri E. Nesterov. "A method of solving a convex programming problem with convergence rate $\mathcal{O}(\frac{1}{k^2})$". In: *Dokl. Akad. Nauk SSSR* 269.3 (1983), pp. 543–547 (cited on pages 48, 81, 130, 136).

[99] Amir Beck and Marc Teboulle. "A fast iterative shrinkage-thresholding algorithm for linear inverse problems". In: *SIAM J. Imaging Sci.* 2.1 (2009), pp. 183–202 (cited on pages 48, 80, 97, 140).

[100] Tuomo Valkonen. "A primal-dual hybrid gradient method for nonlinear operators with applications to MRI". In: *Inverse Problems* 30.5 (2014), pp. 055012, 45 (cited on page 50).

[101] Antonin Chambolle, Matthias J. Ehrhardt, Peter Richtárik, and Carola-Bibiane Schönlieb. "Stochastic primal-dual hybrid gradient algorithm with arbitrary sampling and imaging applications". In: *SIAM J. Optim.* 28.4 (2018), pp. 2783–2808 (cited on page 50).

[102] Yura Malitsky and Thomas Pock. "A first-order primal-dual algorithm with linesearch". In: *SIAM J. Optimiz.* 28.1 (2018), pp. 411–432 (cited on pages 50, 103, 105).

[103] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255 (cited on page 50).

[104] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. "Microsoft coco: Common objects in context". In: *European Conference on Computer Vision*. Springer. 2014, pp. 740–755 (cited on pages 50, 93).

[105] Dimitri P. Bertsekas. *Nonlinear programming*. Third. Athena Scientific, Belmont, MA, 2016 (cited on page 50).

[106] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996 (cited on page 50).

[107] Suvrit Sra. "Scalable nonconvex inexact proximal splitting". In: *Advances in Neural Information Processing Systems*. 2012, pp. 530–538 (cited on pages 50, 51, 93).

[108] Olvi L. Mangasarian and Mikhail V. Solodov. "Backpropagation convergence via deterministic nonmonotone perturbed minimization". In: *Advances in Neural Information Processing Systems*. 1994, pp. 383–390 (cited on page 50).

[109] Angelia Nedić and Dimitri Bertsekas. "Convergence rate of incremental subgradient algorithms". In: *Stochastic optimization: algorithms and applications*. Vol. 54. Appl. Optim. Kluwer Acad. Publ., Dordrecht, 2001, pp. 223–264 (cited on pages 50, 92).

[110] Diederik P. Kingma and Jimmy L. Ba. "ADAM: a method for stochastic optimization". In: *International Conference on Learning Representations*. 2015 (cited on pages 51, 104, 121, 141).

[111] Boris T. Polyak. "Some methods of speeding up the convergence of iterative methods". In: *Ž. Vyčisl. Mat i Mat. Fiz.* 4 (1964), pp. 791–803 (cited on page 51).

[112] Sahank J. Reddi, Satyen Kale, and Sanjiv Kumar. "On the convergence of Adam and beyond". In: *International Conference on Learning Representations*. 2018 (cited on page 51).

[113] David Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2014 (cited on pages 52, 53, 57, 58).

[114] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, 2009 (cited on page 52).

[115] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016 (cited on pages 52, 59–61, 63–66).

[116] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018 (cited on page 52).

[117] Alexander Effland, Erich Kobler, Anne Brandenburg, Teresa Klatzer, Leonie Neuhäuser, Michael Hölzel, Jennifer Landsberg, Thomas Pock, and Martin Rumpf. "Joint reconstruction and classification of tumor cells and cell interactions in melanoma tissue sections with synthesized training data". In: *Int. J. Comput. Assist. Radiol. Surg.* 14.4 (2019), pp. 587–599 (cited on pages 54, 147).

[118] Verma Ruchika, Neeraj Kumar, Abhijee Patil, Nikhil Kurian, Swapnil Rane, and Amit Sethi. "Multi-organ Nuclei Segmentation and Classification Challenge 2020". In: (2020) (cited on page 54).

[119] Kenneth J. Hunt, D. Sbarbaro, R. Żbikowski, and Peter J. Gawthrop. "Neural networks for control systems–a survey". In: *Automatica* 28.6 (1992), pp. 1083–1112 (cited on page 55).

[120] Kerstin Hammernik, Teresa Klatzer, Erich Kobler, Michael P. Recht, Daniel K. Sodickson, Thomas Pock, and Florian Knoll. "Learning a variational network for reconstruction of accelerated MRI data". In: *Magn. Reson. Med.* 79.6 (2018), pp. 3055–3071 (cited on pages 55, 91, 102, 127, 147).

[121] Berthold KP Horn and Brian G Schunck. "Determining optical flow". In: *Techniques and Applications of Image Understanding*. Vol. 281. 1981, pp. 319–331 (cited on page 56).

[122] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. "A naturalistic open source movie for optical flow evaluation". In: *European Conference on Computer Vision*. 2012, pp. 611–625 (cited on page 56).

[123] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. "Flownet: Learning optical flow with convolutional networks". In: *IEEE International Conference on Computer Vision*. 2015, pp. 2758–2766 (cited on page 56).

[124] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8934–8943 (cited on page 56).

[125] Zachary Teed and Jia Deng. "RAFT: Recurrent All-Pairs Field Transforms for Optical Flow". In: *arXiv* 2003.12039 (2020) (cited on pages 56, 145).

[126] Ronald A. Fisher. "The use of multiple measurements in taxonomic problems". In: *Ann. Eugen.* 7.2 (1936), pp. 179–188 (cited on pages 56, 57).

[127] Stuart Lloyd. "Least squares quantization in PCM". In: *IEEE Trans. Inf. Theory* 28.2 (1982), pp. 129–137 (cited on pages 56, 57).

[128] James MacQueen. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. 1967, pp. 281–297 (cited on pages 56, 57).

[129] Olivier Bousquet and André Elisseeff. "Stability and generalization". In: *J. Mach. Learn. Res.* 2.3 (2002), pp. 499–526 (cited on pages 59, 134).

[130] Frank Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain". In: *Psychol. Rev.* 65.6 (1958), p. 386 (cited on page 61).

[131] Robert M. Gray. *Toeplitz and circulant matrices: A review*. Now Publishers Inc., 2006 (cited on page 62).

[132] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. "Semantic image segmentation with deep convolutional nets and fully connected crfs". In: *International Conference on Learning Representations*. 2014 (cited on page 62).

[133] Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. "What is the best multi-stage architecture for object recognition?" In: *International Conference on Computer Vision*. 2009, pp. 2146–2153 (cited on page 63).

[134] Vinod Nair and Geoffrey E Hinton. "Rectified linear units improve restricted Boltzmann machines". In: *International Conference on Machine Learning*. 2010 (cited on page 63).

[135] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. "Deep sparse rectifier neural networks". In: *International Conference on Artificial Intelligence and Statistics*. 2011, pp. 315–323 (cited on page 63).

[136] Sheng Chen, Colin F.N. Cowan, and Peter M. Grant. "Orthogonal least squares learning algorithm for radial basis function networks". In: *IEEE Trans. Neural Netw. Learn. Syst.* 2.2 (1991), pp. 302–309 (cited on page 63).

[137] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *International Conference on Machine Learning*. 2015, pp. 448–456 (cited on page 64).

[138] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. "Instance normalization: The missing ingredient for fast stylization". In: *arXiv* 1607.08022 (2016) (cited on page 64).

[139] Yuxin Wu and Kaiming He. "Group normalization". In: *European Conference on Computer Vision*. 2018, pp. 3–19 (cited on page 64).

[140] Ken-Ichi Funahashi. "On the approximate realization of continuous mappings by neural networks". In: *Neural networks* 2.3 (1989), pp. 183–192 (cited on page 65).

[141] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3431–3440 (cited on page 65).

[142] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising". In: *IEEE Trans. on Image Processing* 26.7 (2017), pp. 3142–3155 (cited on pages 65, 85, 91, 125, 129).

[143] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems*. 2012, pp. 1097–1105 (cited on pages 65, 91, 93).

[144] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning Representations by Back-propagating Errors". In: *Nature* 323.6088 (1986), pp. 533–536 (cited on pages 65, 91).

[145] Hava T. Siegelmann and Eduardo D. Sontag. "On the computational power of neural nets". In: *J. Comput. Syst. Sci.* 50.1 (1995), pp. 132–150 (cited on page 65).

[146] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural Comput.* 9.8 (1997), pp. 1735–1780 (cited on page 66).

[147] Alex Graves, Greg Wayne, and Ivo Danihelka. "Neural turing machines". In: *arXiv* 1410.5401 (2014) (cited on page 66).

[148] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. "Sequence to sequence learning with neural networks". In: *Advances in Neural Information Processing Systems*. 2014, pp. 3104–3112 (cited on page 66).

[149] Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. "Hybrid speech recognition with deep bidirectional LSTM". In: *IEEE Workshop on Automatic Speech Recognition and Understanding*. 2013, pp. 273–278 (cited on page 66).

[150] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 770–778 (cited on pages 66, 90–93).

[151] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Identity mappings in deep residual networks". In: *European Conference on Computer Vision*. 2016, pp. 630–645 (cited on page 66).

[152] E Weinan. "A proposal on machine learning via dynamical systems". In: *Commun. Math. Stat.* 5.1 (2017), pp. 1–11 (cited on page 66).

[153] Eldad Haber and Lars Ruthotto. "Stable architectures for deep neural networks". In: *Inverse Problems* 34.1 (2018), pp. 014004, 22 (cited on pages 66, 113).

[154] Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. "Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations". In: *International Conference on Machine Learning*. 2018, pp. 3276–3285 (cited on page 66).

[155] Bo Chang, Lili Meng, Eldad Haber, Lars Ruthotto, David Begert, and Elliot Holtham. "Reversible Architectures for Arbitrarily Deep Residual Neural Networks". In: *AAAI Conference on Artificial Intelligence*. 2018 (cited on page 66).

[156] Martin Benning, Elena Celledoni, Matthias J Ehrhardt, Brynjulf Owren, and Carola-Bibiane Schönlieb. "Deep learning as optimal control problems: models and numerical methods". In: *arXiv* 1904.05657 (2019) (cited on pages 66, 113).

[157] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. "Densely connected convolutional networks". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 4700–4708 (cited on pages 66, 67).

[158] Arthur Stanley Eddington. "The constants of nature". In: *JR Newman:"The World of Mathematics* 2 (1956), pp. 1074–1093 (cited on page 69).

[159] David H. Hubel and Torsten N. Wiesel. "Receptive fields of single neurones in the cat's striate cortex". In: *J. Physiol.* 148 (1959), pp. 574–591 (cited on page 72).

[160] David H. Hubel and Torsten N. Wiesel. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex". In: *J. Physiol.* 160 (1962), pp. 106–154 (cited on page 72).

[161] Ulf Grenander and Anuj Srivastava. "Probability models for clutter in natural images". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 23.4 (2001), pp. 424–429 (cited on page 73).

[162] Johannes Burge and Wilson S Geisler. "Optimal defocus estimation in individual natural images". In: *Proceedings of the National Academy of Sciences*. 2011, pp. 16849–16854 (cited on pages 73, 76, 81, 91).

[163] Ann B Lee, Kim S Pedersen, and David Mumford. "The Nonlinear Statistics of High-Contrast Patches in Natural Images". In: *Int. J. Comput. Vis.* 54.5413 (2003), pp. 83–103 (cited on page 74).

[164] John Makhoul. "A fast cosine transform in one and two dimensions". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 28.1 (1980), pp. 27–34 (cited on page 74).

[165] Nasir Ahmed, T. Natarajan, and Kamisetty R. Rao. "Discrete cosine transform". In: *IEEE Trans. Comput.* 100.1 (1974), pp. 90–93 (cited on page 75).

[166] Ali Rahimi and Benjamin Recht. "Random features for large-scale kernel machines". In: *Advances in Neural Information Processing Systems*. 2008, pp. 1177–1184 (cited on page 75).

[167] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *International Conference on Artificial Intelligence and Statistics*. 2010, pp. 249–256 (cited on page 75).

[168] Jinggang Huang. “Statistics of Natural Images and Models”. PhD thesis. Brown University, 2000 (cited on page 77).

[169] Daniel L. Ruderman. “Origins of scaling in natural images”. In: *Vis. Res.* 37.23 (1997), pp. 3385–3398 (cited on page 77).

[170] G. J. Burton, Nigel D. Haig, and Ian R. Moorhead. “A self-similar stack model for human and machine vision”. In: *Biological Cybernetics* 53.6 (1986), pp. 397–403 (cited on page 77).

[171] Antonio Turiel, Germán Mato, Néstor Parga, and Jean-Pierre Nadal. “Self-similarity properties of natural images resemble those of turbulent flows”. In: *Phys. Rev. Lett.* 80.5 (1998), p. 1098 (cited on page 77).

[172] Andrey N. Tikhonov and Vasiliy Y. Arsenin. *Solutions of ill-posed problems*. V. H. Winston & Sons, Washington, D.C.: John Wiley & Sons, New York-Toronto, Ont.-London, 1977 (cited on page 77).

[173] Stephen M. Stigler. “Gauss and the invention of least squares”. In: *Ann. Stat.* (1981), pp. 465–474 (cited on page 78).

[174] Leonid I. Rudin. “Images, numerical analysis of singularities and shock filters”. PhD thesis. Caltech, C.S. Dept, 1987 (cited on page 78).

[175] Robert Acar and Curtis R. Vogel. “Analysis of bounded variation penalty methods for ill-posed problems”. In: *Inverse Probl.* 10.6 (1994), p. 1217 (cited on page 79).

[176] C. R. Vogel and M. E. Oman. “Iterative methods for total variation denoising”. In: vol. 17. 1. 1994, pp. 227–238 (cited on page 79).

[177] Tony F. Chan, Gene H. Golub, and Pep Mulet. “A nonlinear primal-dual method for total variation-based image restoration”. In: *SIAM J. Sci. Comput.* 20.6 (1999), pp. 1964–1977 (cited on page 79).

[178] Jamylle L. Carter. “Dual Methods for Total Variation-Based Image Restoration”. PhD thesis. University of California Los Angeles, 2001 (cited on page 79).

[179] Antonin Chambolle. “An algorithm for total variation minimization and applications”. In: vol. 20. 1-2. 2004, pp. 89–97 (cited on page 79).

[180] John van Neumann. “Zur Theorie der Gesellschaftsspiele”. In: *Math. Ann.* 100.1 (1928), pp. 295–320 (cited on page 79).

[181] David C. Dobson and Fadil Santosa. “Recovery of blocky images from noisy and blurred data”. In: *SIAM J. Appl. Math.* 56.4 (1996), pp. 1181–1198 (cited on page 80).

[182] Tony Chan, Antonio Marquina, and Pep Mulet. “High-order total variation-based image restoration”. In: *SIAM J. Sci. Comput.* 22.2 (2000), pp. 503–516 (cited on page 80).

[183] Albert E. Beaton and John W. Tukey. “The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data”. In: *Technometrics* 16.2 (1974), pp. 147–185 (cited on page 81).

[184] Frank R. Hampel, Elvezio M. Ronchetti, Peter J. Rousseeuw, and Werner A. Stahel. *Robust statistics*. John Wiley & Sons, Inc., New York, 1986 (cited on page 81).

[185] Dilip Krishnan and Rob Fergus. “Fast image deconvolution using hyper-Laplacian priors”. In: *Advances in Neural Information Processing Systems*. 2009, pp. 1033–1041 (cited on page 81).

[186] Michael J. Black and Paul Anandan. “The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields”. In: *Comput. Vis. Image Underst.* 63.1 (1996), pp. 75–104 (cited on pages 81, 91).

[187] Yair Weiss and William T Freeman. "What makes a good model of natural images?" In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2007, pp. 1–8 (cited on page 82).

[188] Daniel Zoran and Yair Weiss. "From learning models of natural image patches to whole image restoration". In: *IEEE International Conference on Computer Vision*. 2011, pp. 479–486 (cited on page 82).

[189] Martin Benning, Guy Gilboa, Joana S. Grah, and Carola-Bibiane Schönlieb. "Learning filter functions in regularisers by minimising quotients". In: *International Conference on Scale Space and Variational Methods in Computer Vision*. 2017, pp. 511–523 (cited on page 82).

[190] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. "Optimization by simulated annealing". In: *Science* 220.4598 (1983), pp. 671–680 (cited on page 82).

[191] Magnus R. Hestenes and Eduard Stiefel. "Methods of conjugate gradients for solving linear systems". In: *J. Research Nat. Bur. Standards* 49 (1952), pp. 409–436 (cited on page 82).

[192] Uwe Schmidt and Stefan Roth. "Shrinkage fields for effective image restoration". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 2774–2781 (cited on pages 84, 95).

[193] Kyong Hwan Jin, Michael T McCann, Emmanuel Froustey, and Michael Unser. "Deep convolutional neural network for inverse problems in imaging". In: *IEEE Trans. on Image Processing* 26.9 (2017), pp. 4509–4522 (cited on pages 85, 102, 139).

[194] Hu Chen, Yi Zhang, Jiliu Zhou, and Ge Wang. "Deep learning for low-dose CT". In: *Developments in X-Ray Tomography XI*. 2017, p. 103910I (cited on pages 85, 102, 139).

[195] Eunhee Kang, Junhong Min, and Jong Chul Ye. "A deep convolutional neural network using directional wavelets for low-dose X-ray CT reconstruction". In: *Med. Phys.* 44.10 (2017), e360–e375 (cited on pages 85, 139).

[196] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241 (cited on pages 85, 114).

[197] Richard Zhang. "Making Convolutional Networks Shift-Invariant Again". In: *International Conference on Machine Learning*. 2019 (cited on page 86).

[198] Karl Kunisch and Thomas Pock. "A Bilevel Optimization Approach for Parameter Learning in Variational Models". In: *SIAM J. Imaging Sci.* 6 (2013), pp. 938–983 (cited on pages 87, 88, 91).

[199] Sheheryar Mehmood and Peter Ochs. "Automatic Differentiation of Some First-Order Methods in Parametric Optimization". In: *International Conference on Artificial Intelligence and Statistics*. 2020, pp. 1584–1594 (cited on page 89).

[200] Erich Kobler, Matthew J. Muckley, Baiyu Chen, Florain Knoll, Kerstin Hammernik, Thomas Pock, Daniel K. Sodickson, and Ricardo Otazo. "Variational network learning for low-dose CT". In: *Proceedings of the* 5^{th} *CT Meeting*. 2018 (cited on pages 90, 139, 147).

[201] Li Xu, Shicheng Zheng, and Jiaya Jia. "Unnatural ℓ^0 Sparse Representation for Natural Image Deblurring". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 1107–1114 (cited on page 90).

[202] Tony F. Chan and Luminita A. Vese. "Active Contours Without Edges". In: *IEEE Trans. on Image Processing* 10.2 (2001), pp. 266–277 (cited on page 90).

[203] Marcelo Bertalmio, Guillermo Sapiro, and Gregory Randall. "Morphing Active Contours". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 22.7 (2000), pp. 733–737 (cited on page 90).

[204] Daniel Freedman and Tao Zhang. "Active Contours for Tracking Distributions". In: *IEEE Trans. on Image Processing* 13.4 (2004), pp. 518–526 (cited on page 90).

[205] Xiao-Jiao Mao, Chunhua Shen, and Yu-Bin Yang. "Image restoration using convolutional auto-encoders with symmetric skip connections". In: *arXiv* 1606.08921 (2016) (cited on page 90).

[206] Teresa Klatzer, Kerstin Hammernik, Patrick Knöbelreiter, and Thomas Pock. "Learning joint demosaicing and denoising based on sequential energy minimization". In: *IEEE International Conference on Computational Photography*. 2016, pp. 1–11 (cited on page 91).

[207] Wei Yu, Stefan Heber, and Thomas Pock. "Learning Reaction-Diffusion Models for Image Inpainting". In: *German Conference on Pattern Recognition*. 2015, pp. 356–367 (cited on page 91).

[208] Yann LeCun, Bernhard Boser, John S. Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. "Backpropagation applied to handwritten zip code recognition". In: *Neural computation* 1.4 (1989), pp. 541–551 (cited on page 91).

[209] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. "Deepfool: a simple and accurate method to fool deep neural networks". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 2574–2582 (cited on pages 91, 109).

[210] Andreas Veit, Michael J. Wilber, and Serge Belongie. "Residual networks behave like ensembles of relatively shallow networks". In: *Advances in Neural Information Processing Systems*. 2016, pp. 550–558 (cited on page 93).

[211] Boulch Alexandre. "ShaResNet: reducing residual network parameter number by sharing weights". In: *arXiv* 1702.08782 (2017) (cited on page 93).

[212] Abdi Masoud and Nahavandi Saeid. "Multi-Residual Networks". In: *arXiv* 1609.05672 (2016) (cited on page 93).

[213] Qi Shan, Jiaya Jia, and Aseem Agarwala. "High-quality motion deblurring from a single image". In: *ACM Transs on Graphics* 27.3 (2008), pp. 1–10 (cited on page 94).

[214] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. "Loss functions for neural networks for image processing". In: *arXiv* 1511.08861 (2015) (cited on page 95).

[215] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics". In: *IEEE International Conference on Computer Vision*. 2001, pp. 416–423 (cited on pages 97, 121).

[216] Kevin Schelten, Sebastian Nowozin, Jeremy Jancsary, Carsten Rother, and Stefan Roth. "Interleaved regression tree field cascades for blind image deconvolution". In: *IEEE Winter Conference on Applications of Computer Vision*. 2015, pp. 494–501 (cited on page 97).

[217] Cynthia H. McCollough, Michael R. Bruesewitz, and James JM. Kofler. "CT Dose Reduction and Dose Management Tools: Overview of Available Options". In: *Radiographics* 26.2 (2006), pp. 503–512 (cited on page 102).

[218] Katharine Grant and Rainer Raupach. "SAFIRE: Sinogram Affirmed Iterative Reconstruction". In: *Siemens Medical Solutions Whitepaper* (2012) (cited on pages 102, 105, 107).

[219] Guang-Hong Chen, Jie Tang, and Shuai Leng. "Prior image constrained compressed sensing (PICCS): A method to accurately reconstruct dynamic CT images from highly undersampled projection data sets". In: *Med. Phys.* 35.2 (2008), pp. 660–663 (cited on pages 102, 126).

[220] Yo Seob Han and Jong Chul Ye. "Deep Residual Learning Approach for Sparse-view CT Reconstruction". In: *International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine*. 2017 (cited on page 102).

[221] Lifeng Yu, Maria Shiung, Dayna Jondal, and Cynthia H McCollough. "Development and validation of a practical lower-dose-simulation tool for optimizing computed tomography scan protocols". In: *J. Comput. Assist. Tomogr.* 36.4 (2012), pp. 477–487 (cited on page 105).

[222] Matthew Muckley, Baiyu Chen, Thomas Vahle, Aaron Sodickson, Florian Knoll, Danieln Sodickso, and Ricardo Otazo. "Regularizer Performance for SparseCT Image Reconstruction With Practical Subsampling". In: *International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine*. 2017 (cited on page 105).

[223] Baiyu Chen, M. Muckley, T. O'Donnell, A. Sodickson, T. Flohr, K. Stierstorfer, B. Schmidt, F. Knoll, A. Primak, D. Faul, D. Sodickson, and R. Otazo. "Realistic Undersampling Model for Compressed Sensing Using a Multi-Slit Collimator". In: *International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine*. 2017 (cited on page 106).

[224] Erich Kobler, Baiyu Chen, Alexander Effland, Thomas Pock, Daniel K. Sodickson, and Ricardo Otazo. "Total Deep Variation for SparseCT Reconstruction". In: *Proceedings of the 6^{th} CT Meeting*. 2020 (cited on pages 108, 148).

[225] Mark Nitzberg, David Mumford, and Takahiro Shiota. *Filtering, segmentation and depth*. Vol. 662. Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1993 (cited on page 109).

[226] Tony F. Chan, Sung Ha Kang, and Jianhong Shen. "Euler's elastica and curvature-based inpainting". In: *SIAM J. Appl. Math.* 63.2 (2002), pp. 564–592 (cited on page 109).

[227] Antonin Chambolle and Thomas Pock. "Total roto-translational variation". In: *Numer. Math.* 142.3 (2019), pp. 611–666 (cited on page 109).

[228] Anat Levin and Boaz Nadler. "Natural image denoising: Optimality and inherent bounds". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2011, pp. 2833–2840 (cited on page 109).

[229] Stamatios Lefkimmiatis. "Non-local Color Image Denoising with Convolutional Neural Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 5882–5891 (cited on page 109).

[230] Vegard Antun, Francesco Renna, Clarice Poon, Ben Adcock, and Anders C. Hansen. "On instabilities of deep learning in image reconstruction and the potential costs of AI". In: *Proceedings of the National Academy of Sciences* (2020). DOI: 10.1073/pnas.1907377117 (cited on page 109).

[231] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. "Intriguing properties of neural networks". In: *International Conference on Learning Representations*. 2014 (cited on page 109).

[232] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *International Conference on Learning Representations*. 2015 (cited on page 109).

[233] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. "Universal adversarial perturbations". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1765–1773 (cited on page 109).

[234] Andrew Y. Ng and Michael I. Jordan. "On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes". In: *Advances in Neural Information Processing Systems 14*. MIT Press, 2002, pp. 841–848 (cited on page 110).

[235] Heinz W. Engl, Martin Hanke, and Andreas Neubauer. *Regularization of inverse problems*. Vol. 375. Mathematics and its Applications. Kluwer Academic Publishers Group, Dordrecht, 1996 (cited on page 110).

[236] A. N. Tikhonov, A. S. Leonov, and A. G. Yagola. *Nonlinear ill-posed problems*. Vol. 14. Applied Mathematics and Mathematical Computation. Chapman & Hall, London, 1998 (cited on page 110).

[237] Andreas Rieder. *Keine Probleme mit inversen Problemen*. Friedr. Vieweg & Sohn, Braunschweig, 2003 (cited on page 110).

[238] Louis Landweber. "An Iteration Formula for Fredholm Integral Equations of the First Kind". In: *American Journal of Mathematics* 73.3 (1951), pp. 615–624 (cited on page 110).

[239] Y. Zhang and B. Hofmann. "On the second-order asymptotical regularization of linear ill-posed inverse problems". In: *Appl. Anal.* 99.6 (2020), pp. 1000–1025 (cited on page 110).

[240] Garvesh Raskutti, Martin J. Wainwright, and Bin Yu. "Early stopping and non-parametric regression: an optimal data-dependent stopping rule". In: *J. Mach. Learn. Res.* 15.1 (2014), pp. 335–366 (cited on page 110).

[241] Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. "On early stopping in gradient descent learning". In: *Constr. Approx.* 26.2 (2007), pp. 289–315 (cited on page 110).

[242] A. Binder, M. Hanke, and O. Scherzer. "On the Landweber iteration for nonlinear ill-posed problems". In: vol. 4. 5. 1996, pp. 381–389 (cited on page 110).

[243] Tong Zhang and Bin Yu. "Boosting with early stopping: convergence and consistency". In: *Ann. Statist.* 33.4 (2005), pp. 1538–1579 (cited on page 110).

[244] Lutz Prechelt. "Early Stopping—But When?" In: *Neural Networks: Tricks of the Trade*. Springer, 2012, pp. 53–67 (cited on page 110).

[245] Lorenzo Rosasco and Silvia Villa. "Learning with incremental iterative regularization". In: *Advances in Neural Information Processing Systems*. 2015, pp. 1630–1638 (cited on page 111).

[246] Simon Matet, Lorenzo Rosasco, Silvia Villa, and Bang Long Vu. "Don't relax: early stopping for convex regularization". In: *arXiv* 1707.05422 (2017) (cited on page 111).

[247] Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient flows in metric spaces and in the space of probability measures*. Second. Lectures in Mathematics ETH Zürich. Birkhäuser Verlag, Basel, 2008 (cited on page 113).

[248] Eberhard Zeidler. *Nonlinear functional analysis and its applications. III*. Springer-Verlag, New York, 1985 (cited on page 117).

[249] Xixi Jia, Sanyang Liu, Xiangchu Feng, and Lei Zhang. "FOCNet: A Fractional Optimal Control Network for Image Denoising". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 6047–6056 (cited on page 125).

[250] Kai Zhang, Wangmeng Zuo, and Lei Zhang. "FFDNet: Toward a Fast and Flexible Solution for CNN-Based Image Denoising". In: *IEEE Trans. on Image Processing* 27.9 (2018), pp. 4608–4622 (cited on page 125).

[251] Tobias Plötz and Stefan Roth. "Neural Nearest Neighbors Networks". In: *Advances in Neural Information Processing Systems 31*. Curran Associates, Inc., 2018, pp. 1087–1098 (cited on page 125).

[252] Sungsoo Ha and Klaus Mueller. "A Look-Up Table-Based Ray Integration Framework for 2-D/3-D Forward and Back Projection in X-Ray CT". In: *IEEE Trans. on Med. Imaging* 37 (2018), pp. 361–371 (cited on pages 125, 139).

[253] Ningning Zhao, Qi Wei, Adrian Basarab, Nicolas Dobigeon, Denis Kouamé, and Jean-Yves Tourneret. "Fast single image super-resolution using a new analytical solution for ℓ_2–ℓ_2 problems". In: *IEEE Trans. Image Process.* 25.8 (2016), pp. 3683–3697. DOI: 10.1109/TIP.2016.2567075 (cited on page 128).

[254] Xiangyu He, Zitao Mo, Peisong Wang, Yang Liu, Mingyuan Yang, and Jian Cheng. "ODE-inspired Network Design for Single Image Super-Resolution". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 1732–1741 (cited on pages 128, 129).

[255] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. "MemNet: A Persistent Memory Network for Image Restoration". In: *IEEE International Conference on Computer Vision*. 2017, pp. 4549–4557 (cited on page 129).

[256] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. "Accurate Image Super-Resolution Using Very Deep Convolutional Networks". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 1646–1654 (cited on page 129).

[257] Ying Tai, Jian Yang, and Xiaoming Liu. "Image Super-Resolution via Deep Recursive Residual Network". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 2790–2798 (cited on page 129).

[258] Guy Gilboa. *Nonlinear eigenproblems in image processing and computer vision*. Advances in Computer Vision and Pattern Recognition. Springer, Cham, 2018 (cited on page 130).

[259] Baiyu Chen, Erich Kobler, Matthew J. Muckley, Aaron D. Sodickson, Thomas O'Donnell, Thomas Flohr, Bernhard Schmidt, Daniel K. Sodickson, and Ricardo Otazo. "SparseCT: System concept and design of multislit collimators". In: *Med. Phys.* 46.6 (2019), pp. 2589–2599 (cited on pages 138, 148).

[260] Yoseob Han and Jong Chul Ye. "Framing U-Net via deep convolutional framelets: Application to sparse-view CT". In: *IEEE Trans. Med. Imaging* 37.6 (2018), pp. 1418–1429 (cited on page 139).

[261] Andreas Kofler, Markus Haltmeier, Tobias Schaeffter, Marc Kachelrieß, Marc Dewey, Christian Wald, and Christoph Kolbitsch. "Neural networks-based regularization for large-scale medical image reconstruction". In: *Phys. Med. Biol.* 65.13 (2020), p. 135003 (cited on page 139).

[262] Donghwan Kim, Sathish Ramani, and Jeffrey A. Fessler. "Combining Ordered Subsets and Momentum for Accelerated X-Ray CT Image Reconstruction". In: *IEEE Trans. Med. Imaging* 34.1 (2015), pp. 167–178 (cited on page 140).

[263] Ana F. Vidal, Valentin De Bortoli, Marcelo Pereyra, and Alain Durmus. "Maximum likelihood estimation of regularisation parameters in high-dimensional inverse problems: an empirical Bayesian approach". In: *arXiv* 1911.11709 (2019) (cited on page 145).

[264] Jonas Adler and Ozan Öktem. "Deep bayesian inversion". In: *arXiv* 1811.05910 (2018) (cited on page 145).

[265] Teresa Klatzer, Daniel Soukup, Erich Kobler, Kerstin Hammernik, and Thomas Pock. "Trainable regularization for multi-frame superresolution". In: *German Conference on Pattern Recognition*. 2017, pp. 90–100 (cited on page 147).

[266] Alexander Effland, Michael Hölzel, Teresa Klatzer, Erich Kobler, Jennifer Landsberg, Leonie Neuhäuser, Thomas Pock, and Martin Rumpf. "Variational Networks for Joint Image Reconstruction and Classification of Tumor Immune Cell Interactions in Melanoma Tissue Sections". In: *Bildverarbeitung für die Medizin*. 2018, pp. 334–340 (cited on page 147).

[267] Kerstin Hammernik, Erich Kobler, Thomas Pock, Michael P. Recht, Daniel K. Sodickson, and Florian Knoll. "Variational adversarial networks for accelerated MR image reconstruction". In: *Proceedings of the International Society of Magnetic Resonance in Medicine*. 2018 (cited on page 147).

[268] Florian Knoll, Kerstin Hammernik, Erich Kobler, Thomas Pock, Daniel K. Sodickson, and Michael P. Recht. "Analysis of the influence of deviations between training and test data in learned image reconstruction". In: *ISMRM Workshop on Machine Learning*. 2018 (cited on page 147).

[269] Florian Knoll, Kerstin Hammernik, Erich Kobler, Thomas Pock, Michael P. Recht, and Daniel K. Sodickson. "Assessment of the generalization of learned image reconstruction and the potential for transfer learning". In: *Magn. Reson. Med.* 81.1 (2019), pp. 116–128 (cited on page 147).

[270] Alexander Effland, Erich Kobler, Thomas Pock, and Martin Rumpf. "Time discrete geodesics in deep feature spaces for image morphing". In: *International Conference on Scale Space and Variational Methods in Computer Vision*. 2019, pp. 171–182 (cited on page 147).

[271] Alexander Effland, Erich Kobler, Thomas Pock, Marko Rajković, and Martin Rumpf. "Image Morphing in Deep Feature Spaces: Theory and Applications". In: *J. Math. Imaging Vision* 62.3 (2020), pp. 396–416 (cited on page 148).

[272] Martin Zach and Erich Kobler. "Real-World Video Restoration using Noise-2-Noise". In: *Proceedings of the Joint Austrian Computer Vision and Robotics Workshop*. 2020 (cited on page 148).

[273] Elena A. Kaye, Emily A. Aherne, Cihan Duzgol, Ida Häggström, Erich Kobler, Yousef Mazaheri, Maggie M. Fung, Zhigang Zhang, Ricardo Otazo, Herbert A. Vargas, and Oguz Akin. "Accelerating Prostate Diffusion Weighted MRI using Guided Denoising Convolutional Neural Network: Retrospective Feasibility Study". In: *Radiology: Artificial Intelligence* 2.5 (2020), e200007 (cited on page 148).

List of Acronyms

A
ADMM alternating direction method of multipliers. 6

C
CG conjugate gradient. 129, 144
CNN convolutional neural network. ix, xi, 6, 66, 87, 88, 117, 118, 142, 143, 147
CS compressed sensing. 3, 105
CT computed tomography. 2, 3, 7, 8, 56, 71, 72, 79, 93, 105–109, 111, 128, 129, 131, 141–144, 146–148

D
DCT discrete cosine transform. 76–78
DenseNet dense neural network. 67, 68

F
FBP filtered back-projection. 142
FISTA fast iterative shrinkage and thresholding algorithm. 49
FoE fields of experts. ix, xi, 5–8, 85–88, 91, 94, 97, 106, 111, 112, 116, 117, 124–126, 128, 133, 147

G
GAN generative adversarial network. 6
GB giga byte. 7
GMM Gaussian mixture model. 57
GPU graphics processing unit. 7, 144

H
HU Hounsfield units. 107, 108

I
i.i.d. independent and identically distributed. 3, 29, 72

L
l.s.c. lower semi-continuous. 95

M
MAP maximum a-posteriori. 5, 6, 58, 59, 71, 72, 90, 111, 148
MOSFET metal oxide semiconductor field-effect transistor. 53
MRF Markov random field. 5, 83, 84
MRI magnetic resonance imaging. 1–3, 56, 71, 79, 111, 130, 131, 148
MSC multislit collimator. 141, 144

N
NN neural network. 8, 53, 61, 62, 64–67

O
ODE ordinary differential equation. 33–36, 67

P
PDHG primal-dual hybrid gradient. 6
PSNR peak signal-to-noise ratio. 114

R
ReLU rectified linear unit. 96

ResNet residual neural network. ix, xi, 7, 8, 67, 68, 147
RKHS reproducing kernel Hilbert spaces. 113
RMSE root mean squared error. 108
RNN recurrent neural network. 6, 66, 67, 106, 111
ROF Rudin-Osher-Fatemi. 93, 94, 101

S
SCT sparse computed tomography. 106, 108
SISR single image super-resolution. 3, 111, 131, 133, 148
SVM support vector machine. 80

T
TCR tube current reduction. 106, 108
TDV total deep variation. ix, xi, 6–8, 87, 88, 91, 111, 113, 116–118, 124–126, 128, 131, 133, 141, 143, 144, 146–148
TGV total generalized variation. 82, 84, 91, 94, 142, 143
TNRD trainable nonlinear reaction diffusion. 6–8, 147
TV total variation. 4, 5, 14, 81–84, 88, 91, 105, 106, 112, 114, 142, 143

V
VM variational model. 93, 94
VN variational network. ix, xi, 6–8, 93–95, 97, 98, 100–109, 111, 112, 147