Technische Universität München
Lehrstuhl für Datenverarbeitung

Model-based learning of co-sparse representations for image processing applications

Martin Kiechle

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzende(r): Prof. Dr.-Ing. Gerhard Rigoll

Prüfer der Dissertation:

1. Priv.-Doz. Dr. rer. nat. Martin Kleinsteuber
2. Prof. Dr.-Ing. Eckehard Steinbach

Die Dissertation wurde am 04.04.2019 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 07.10.2019 angenommen.

Martin Kiechle. *Model-based learning of co-sparse representations for image processing applications.* Dissertation, Technische Universität München, Munich, Germany, 2019.

To my beloved Kat

Our greatest weakness lies in giving up.
The most certain way to succeed is always
to try just one more time.

–THOMAS ALVA EDISON–

Abstract

Highly effective solutions for computer vision and image processing applications require models that characterize well the important aspects of the involved image data. Two paradigms for creating image models have emerged in the literature. One relies on expert knowledge that captures analytical properties of the involved data in handcrafted feature models and requires no annotated data. Data-driven modeling, on the other hand, leverages large amounts of annotated training data to infer descriptive features automatically in fully learned representations. In this thesis, methods are explored that combine advantageous aspects of both paradigms, with the goal of reducing training complexity in comparison to fully learned and improving accuracy over handcrafted models. To that end, analytic knowledge of the image formation in specific applications is formalized into optimality criteria for unsupervised learning tasks, which parameterize low-level representation models from unlabeled training data. This is achieved by extending the well-established sparsity principle with knowledge from several classical image processing problems.

First, the problem of reconstructing partial and corrupted photometric images is considered. Under the assumption that such image data is formed under variations in brightness and contrast, a new co-sparse analysis model and a suitable numerical method are described which allow learning filters that extract local image structure invariantly of changes in illumination. The proposed algorithm exploits the geometric properties of the learning problem by minimizing the associated cost function with a conjugate gradient method on a product of spheres manifold, to find numerical solutions efficiently. Evaluated in a practical task of image reconstruction from partial information, the proposed approach is shown to improve results over existing methods in terms of quality as well as size of the required training set.

In the second part, the study of the reconstruction problem is extended to multi-modal images. There, image data of an environment is acquired simultaneously by different imaging techniques and the task is to reconstruct a high-quality version of one image by leveraging the others. To this end, a new model is introduced, that captures the essential structure of photometric image data in the optical range and its interdependency with near-infrared and depth camera data. Further, it is assumed that an environment induces dependent patterns in different image modalities recorded by a bimodal camera setup. A pair of two filter sets is learned, such that aligned and co-occurring patterns are modeled through coupled representations. The derived learning function augments the co-sparsity objective to

select representations that are simultaneously sparse and whose patterns are correlated. To derive optimal filter sets from a small number of training samples, the unimodal numerical algorithm is extended for this multi-modal setup. Subsequently, the parameterized model is employed in a novel image reconstruction algorithm, where a high-quality photometric image is used to reconstruct an aligned depth map from few noisy measurements and achieves state-of-the-art results. Furthermore, image registration is studied as another application for this image model. Utilizing that the introduced bimodal co-sparse analysis model captures interdependent local structures of aligned images, a new image registration algorithm is introduced, which estimates the parameters of a rigid transformation between unaligned images from different modalities. The task is cast as an optimization problem over Lie groups, and its data term relies on the features extracted by the learned bimodal analysis operators. Its results compare favorably with prior methods on different pairs of image modalities.
In the third part, unsupervised texture segmentation is investigated. There, an image needs to be partitioned into non-overlapping sections based on differently textured image areas. Features that capture well the textural patterns of a certain class of images are crucial for the performance of texture segmentation methods. The manual selection of features is a tedious task, while automatically finding such requires a large set of training images and ground truth segmentation labels. Here, a framework is proposed to determine such features when no labeled training data is available. The cost function for the optimization procedure augments the co-sparsity objective of the analysis operator learning to match the commonly used piecewise constant Mumford-Shah segmentation model. This means that the representations are learned such that they provide an approximately piecewise constant feature image with a sparse jump set. The corresponding numerical procedure can learn these representations from a small set of images, from a single image, or even from image patches. Finally, a segmentation algorithm is presented that leverages the learned features to produce label maps of texture segments of an image. The results achieved by this model and its segmentation algorithm on well-established benchmark datasets outperform almost all prior methods.

Acknowledgments

This dissertation marks an important milestone of my academic journey and I want to take this opportunity to thank all of those great individuals who have supported me on this path. Firstly, I would like to express my sincere gratitude to my advisor and mentor Prof. Martin Kleinsteuber. Thanks a lot for guiding me through my dissertation, encouraging me when things seemed impossible, supporting me in the elaboration of great ideas through constructive discussions, and connecting me with bright minds in the community. Your enthusiasm for science, humorous and open nature made my research time at TUM very enjoyable. Special thanks go to Simon, who initially sparked my enthusiasm for this topic and introduced me to this project. My appreciation also extends to Tim, Andreas, Martin and Dima for the great collaboration on joint publications. Your insights and constructive feedback have been central to my own learning and you made the hard work fun. Thanks to Clemens, Julian, Dominik, Hao, Matthias, Uli and all colleagues at LDV for all the fruitful discussions and for making this time so memorable. A big thank-you goes to the Cobrainer team. They often had to forgo my expertise and tolerate my absence, but never complained. Thank you for your understanding and support. I dearly thank my family. They have given me a lot of tailwind and never lost faith in me. I consider myself very lucky to have such a family. Above all, I thank my wife Katharina for her love and constant support, for all the late nights and early mornings, and for always providing me direction. Thank you for being my muse, editor, proofreader, and sounding board. But most of all, thank you for being my best friend.

Contents

List of Figures

List of Tables

List of Symbols

$\mathrm{X}, \mathrm{Y}, \mathrm{Z}$	Sets, written as upper-case letters
$\mathcal{X}, \mathcal{Y}, \mathcal{Z}$	Tensors, written as upper-case calligraphic letters
$\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$	Matrices, written as upper-case boldface letters
$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$	Vectors, written as lower-case boldface letters
$\alpha, \beta, x, y, M, N$	Scalars, written as italic letters
$\boldsymbol{x}_j$	j-th column of matrix $\boldsymbol{X}$
$\boldsymbol{X}_{ij}$	Coefficient in i-th row and j-th column of matrix $\boldsymbol{X}$
$x_i, (\boldsymbol{x})_i$	i-th entry of vector $\boldsymbol{x}$
$\mathcal{X}_{:,:,i}$	i-th slice of tensor $\mathcal{X}$
$\boldsymbol{X}^{-1}$	Inverse of matrix $\boldsymbol{X}$
$\boldsymbol{X}^{\dagger}$	Moore-Penrose pseudo-inverse of matrix $\boldsymbol{X}$
$\boldsymbol{X}^{\top}, \boldsymbol{x}^{\top}$	Transpose of matrix, vector
$\boldsymbol{x}^{\star}, \boldsymbol{X}^{\star}$	Optimizer of an optimization problem
$\operatorname{supp}(\boldsymbol{x})$	Support of the vector $\boldsymbol{x}$
$\operatorname{cosupp}(\boldsymbol{x})$	Co-support of the vector $\boldsymbol{x}$
$\boldsymbol{I}, \boldsymbol{I}_n$	Identity matrix of size $\mathbb{R}^{n \times n}$
$\boldsymbol{J}, \boldsymbol{J}_n$	Matrix of size $\mathbb{R}^{n \times n}$ where all coefficients equal 1
$\mathbb{1}, \mathbb{1}_n$	Vector of size $\mathbb{R}^n$ where all coefficients equal 1
$\det \boldsymbol{X}$	Determinant of matrix $\boldsymbol{X}$
$\operatorname{rank}(\boldsymbol{X})$	Rank of matrix $\boldsymbol{X}$
$\operatorname{tr} \boldsymbol{X}$	Trace of matrix $\boldsymbol{X}$
$\operatorname{vec}(\boldsymbol{X})$	Operator that stacks the columns of matrix $\boldsymbol{X}$
$\operatorname{vec}^{-1}(\boldsymbol{x})$	Operator that unstacks the columns of matrix $\boldsymbol{X}$ from vector $\boldsymbol{x}$
$\operatorname{cov}(\mathcal{X})$	Covariance matrix of vectors along the third axis of tensor $\mathcal{X}$
$\langle \cdot, \cdot \rangle$	Standard inner product

$\boldsymbol{X} \odot \boldsymbol{Y}$	Hadamard (element-wise) product of two matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$
$\boldsymbol{X} \otimes \boldsymbol{Y}$	Kronecker product of two matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$
$e^{\boldsymbol{X}}$	Matrix exponential
$\mathrm{OB}(n, k)$	Oblique manifold, product structure of k elements
$\boldsymbol{x}^{\perp}, \boldsymbol{X}^{\perp}$	Orthogonal complement of the vector, matrix
$\mathbb{S}_{n-1}$	Unit sphere in $\mathbb{R}^n$
$\nabla f(\boldsymbol{X})$	Gradient of function f with respect to $\boldsymbol{X}$
$\boldsymbol{\Omega}^F$	Full analysis operator that can be applied to entire images as a convolutional operator

List of Acronyms

ACoSaMP	Analysis CoSaMP
ADMM	Alternating Direction Method of Multipliers
AHTP	Analysis Hard Thresholding Pursuit
AIHT	Analysis Iterative Hard Thresholding
AOLA	Analysis Operator Learning Algorithm
AR3D	3D Auto Regressive Model With EM
ASP	Analysis Subspace Pursuit
ATLAS	Automatically Tuned Linear Algebra Software
BG	Backward Greedy
C-GOAL	GeOmetric Analysis operator Learning with Centered rows
CAOL	Constrained Analysis Operator Learning
CG	Conjugate Gradient
CoSaMP	Compressive Sampling Matching Pursuit
CUDA	Compute Unified Device Architecture
DSLR	Digital Single Lens Reflex
EM	Expactation Maximization
FSEG	Factorization-Based Texture Segmentation
GAP	Greedy Analysis Pursuit
GCE	Global Consistency Error
GMRF	Gaussian Markov Random Field

GOAL	GeOmetric Analysis operator Learning
GRASP	GReedy Analysis Structured Pursuit
HR	High-Resolution
HTP	Hard Thresholding Pursuit
IHT	Iterative Hard Thresholding
JBAO	Joint Bimodal Analysis Operator
LiDaR	Light Detection and Ranging
LR	Low-Resolution
M3C-JAOL	Multi-Modal Multi-Channel Joint Analysis Operator Learning
MATLAB	MATrix LABoratory
MI	Mutial Information
MKL	Math Kernel Library
MRF	Markov Random Fields
NIR	Near-Infrared
NMI	Normalized Mutial Information
NP	Non-deterministic polynomial-time
OB	Oblique Manifold
OBG	Optimized Backward Greedy
OMP	Orthogonal Matching Pursuit
OpenBLAS	Open Basic Linear Algebra Subroutines
PCA	Principal Component Analysis
PCA-MS	Variational Multi-Phase Segmentation
PMCFA	Priority Multi Class Flooding Algorithm
PSNR	Peak Signal-to-Noise Ratio
RGB	Red-Green-Blue
RGB-D	Red-Green-Blue-Depth

RMSE	Root-Mean-Squared Error
RS	Regression-based Segmenation
SSIM	Structural SIMilarity
SP	Subspace Pursuit
SR	Super-Resolution
SVD	Singular Value Decomposition
SWA	Segmentation by Weighted Aggregation
TFR	Texture Fragmentation and Reconstruction
TS	Texel-based Segmentation
TV	Total Variation
UNTF	Uniform Normalized Tight Frame

Chapter 1

Introduction

Humans are visual creatures that often rely primarily on their visual perception in decision-making and visual stimuli are known to sometimes even alter the judgment of input from other senses [27, 116]. It is therefore unsurprising that technical solutions that mimic the human visual system have a long history. It extends from early analog photography over digital imaging to computer vision and artificial image understanding. Devices that record still and motion pictures have long been a commodity and have helped to analyze problems, communicate visually with peers, observe the environment or to simply memorize important life events. The trend of digitizing more and more visual information has grown at an unprecedented rate and the amount of available digital image data has become truly overwhelming. Making sense of this data in automated and scalable ways has the potential to help developing a better understanding of nature and to unlock numerous new applications across all industries. The cognitive processes that allow humans to make sense of information contained in visual data involve steps to map the physical sensation in visual receptors to previously established models of the world. Technical systems are designed to mimic this behavior and transform raw input data to representations that capture the relevant information for decision making and action. Along this transition, the information is transformed through multiple steps, where robust representations of low-level visual objects are required initially to enable subsequent processing steps on more complex objects of higher-level abstraction [157].

This thesis concerns itself with several low-level processing steps of an artificial visual cognition system. First, it addresses the need to obtain a robust representation of image structure

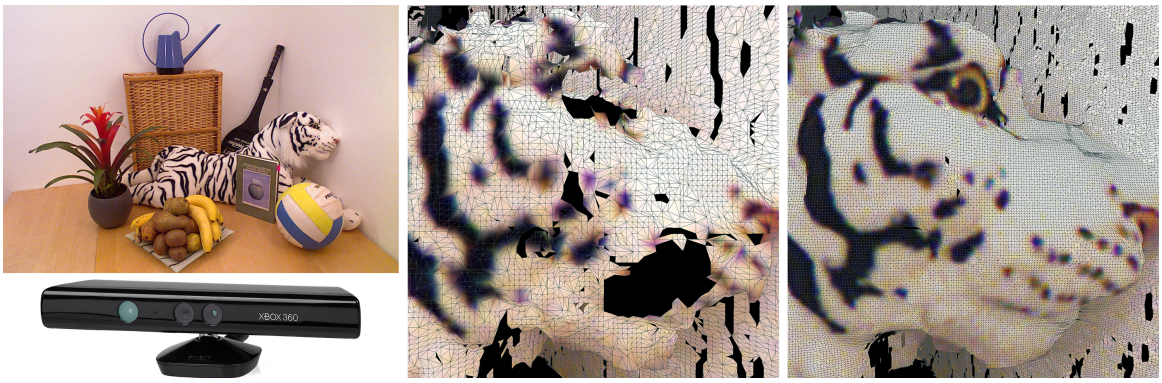


Figure 1.1: Sample bimodal image reconstruction result. Data from a bimodal image sensor that captures optical light and scene depth is enhanced by modeling representation of bimodal image structure on the super-pixel level. *Bottom left:* Microsoft Kinect sensor [41]. *Top left:* scene captured by the color camera. *Middle:* 3D rendering of the original corrupted sensor data. *Right:* data reconstructed by the proposed method from Chapter 4.

on pixel and super-pixel level. Methods are developed, which allow a technical system to establish robust low-level visual representations with the goal of reconstructing erroneous and incomplete sensor data caused by deteriorating effects in the digital image acquisition process. This is crucial for any subsequent higher-level cognitive processing.

Second, the goal of improving low-level visual representations is further pursued by combining data from *multi-modal* image sensors. Multi-modality of the data refers to disparate physical properties of the environment that are captured by different imaging sensor types simultaneously. To that end, an approach for multi-modal representations is proposed which allows to integrate several image sources beneficially, such that errors produced in one sensor can be mitigated by the information captured by another. Figure 1.1 shows an example result of bimodal image reconstruction achieved with the method proposed in Chapter 4: scene depth image data is enhanced with additional scene color images captured by sensor technologies with vastly different noise and resolution characteristics. Furthermore, it is shown that these multi-modal representations also help with spatially misaligned measurements from these sensors.

Third, a technique to establish representations of different *image textures* is described that allows to partition the image plane into segments of different visual objects, known as *image segmentation*. This is useful for the purpose of highlighting areas of interest to a human

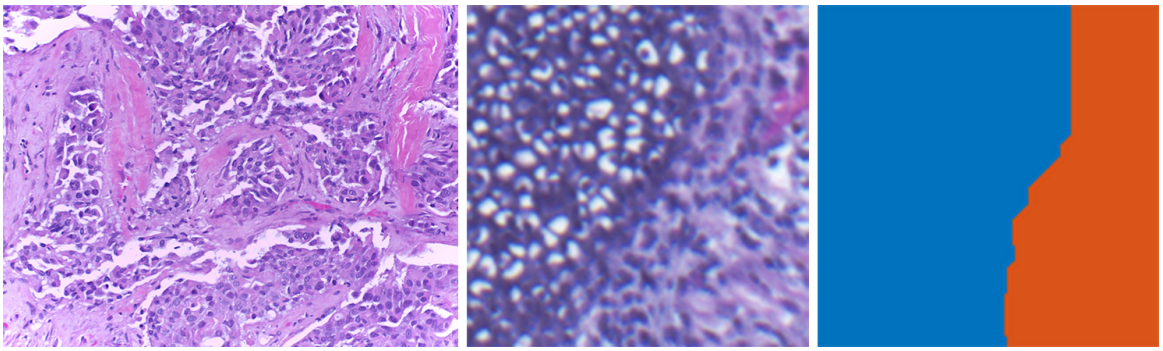


Figure 1.2: Sample unsupervised texture segmentation result. A histology image is automatically divided into partitions representing different tissue segments. Stained tissue image captured by a microscope from the BACH Dataset [13] (left), excerpt of the boundary between two tissue segments (middle) and automatic segmentation achieved by the proposed texture segmentation method (right) in Chapter 5.

observer or to focus subsequent processing steps on a subset of pixels in the input image. Figure 1.2 depicts one sample application, where automatic segmentation of histology images may assist medical diagnostics.

Image reconstruction, alignment and segmentation as inverse problems

Inverse problems refer to problems whose formulation require the result of another problem, often called the *forward* or *direct problem* that is typically oriented along a cause-effect sequence or a loss of information [88, 18]. Three direct problems frequently occur when dealing with digital image processing and are covered in this thesis. The digital image acquisition as the transformation of physical properties through the measurement device and the digitization process forms the first direct problem. It describes erroneous processing steps that introduce a loss of information which is desired to be undone in a corresponding *reconstruction*, constituting the inverse problem. Sensor recordings from multiple imaging devices that map measurements of the same points in the three-dimensional environment to different points in the two-dimensional data, describe the second forward problem. Here, the information of correspondence in the image planes is lost. Spatial *alignment* of these measurements forms the respective inverse problem. The third forward problem is described

by the imaging of three-dimensional objects in a two-dimensional plane. Given that these are non-transparent, pixels that relate to measurements of these objects form non-overlapping contiguous segments. *Segmentation* of the image to recover object boundaries establishes the third inverse problem.

Formalizing the forward and inverse problems, an example from the case of image acquisition and reconstruction is considered as follows. Let $\boldsymbol{x}$ be some signal from a class $\mathrm{X} \subseteq \mathbb{R}^n$ that passes a linear measurement device with system matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$. The direct problem of measuring the signal $\boldsymbol{x}$ under additive noise $\boldsymbol{n}$ then reads as

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{n}. \tag{1.1}$$

Unfortunately it is in general not trivial to recover the original signal $\boldsymbol{x}$ from the measurements $\boldsymbol{y}$. First, this is due to the stochasticity of noise. Even if $\boldsymbol{A}$ is the identity and the measurements $\boldsymbol{y}$ obtained from signal $\boldsymbol{x} \in \mathbb{R}^n$ under additive Gaussian noise are considered, i.e.

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n}, \quad \boldsymbol{n} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}), \tag{1.2}$$

correctly determining the original signal from the measurement is impossible despite having a full statistical description of the noise. Second, even in the absence of noise but with a system matrix $\boldsymbol{A}$ that is different from the identity, the inverse problem may be ill-posed. This means that it doesn't satisfy uniqueness, existence and continuity [18]. For linear operators $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, the problem becomes under- or over-determined depending on the rank of $\boldsymbol{A}$, leading to no or infinite solutions respectively. In order to solve an ill-posed inverse problem, one typically resorts to finding an approximate solution instead that needs to satisfy additional constraints [18]. To that end, first a function is defined

$$f \colon \mathbb{R}^m \to \mathbb{R}_+ \tag{1.3}$$

that measures the deviation of an approximate solution from the observed data. Respectively, a solution to the inverse problem formally needs to satisfy

$$f(\boldsymbol{y}, \boldsymbol{A}\boldsymbol{x}) \leq \varepsilon, \tag{1.4}$$

with ε representing the maximum allowed deviation of the estimate from the observed data.

Since such a solution is not unique in general, one may end up with a result that does not resemble the original signal. To improve on this issue, prior information about the original signal and hence the expected solution is required to constrain the solution space in a meaningful way. Naturally, these constraints should restrict the solution space to those parts, where its elements fall into the expected signal class X and the characteristics that define these parts are often referred to as a *signal model* of X, here in short $\boldsymbol{\mathcal{M}}_{\mathbf{X}}$. In addition to establishing a model, a measure of model error

$$g_{\boldsymbol{\mathcal{M}}} \colon \mathbb{R}^n \to \mathbb{R}_+ \tag{1.5}$$

is required, which indicates if an approximate $\boldsymbol{x}^\star \in \mathbb{R}^n$ shares its characterization with the model of X. The model error can then be used to constrain the inverse problem Eq. (1.4). The objective of finding a solution to the inverse of the direct problem in Eq. (1.1) then amounts to finding a candidate that satisfies model fitness while at the same time being consistent with observations, or more formally

$$\boldsymbol{x}^\star \in \operatorname{argmin} g_{\boldsymbol{\mathcal{M}}}(\boldsymbol{x}) \quad \text{subject to} \quad f(\boldsymbol{y}, \boldsymbol{A}\boldsymbol{x}) \leq \varepsilon. \tag{1.6}$$

From Eq. (1.6) it is clear that the success of finding a high quality solution depends on a good model $\boldsymbol{\mathcal{M}}$ of X. It helps to differentiate whether *any* estimate that is close to its observation resembles an element of X and is therefore a viable solution. As a consequence, a major part of this thesis is dedicated to developing new data models for image data to solve the inverse problems of reconstruction, alignment and segmentation for specific classes of images.

1.1 Image models and representations

A useful model $\boldsymbol{\mathcal{M}}$ captures the key features of a signal class $\mathrm{X} \subseteq \mathbb{R}^n$ in a structure that allows to determine whether any given image is a member of that class or not [34]. As introduced in the previous section, the measure $g_{\boldsymbol{\mathcal{M}}}$ is required to quantify the model fitness. The accuracy and efficiency of determining this fitness are not only influenced by the complexity of features, but also by the *representation* that is chosen for the image data. In this context,

an array of scalars for gray-scale or of vectors for multi-channel images is the most common representation of the spatially discretized version of the continuous two-dimensional function of light intensity. Each cell of the array refers to a pixel in the image and the value represents illumination intensity in the respective (color) channel. The pixel-array of a single-channel image is referred to as $\boldsymbol{x} = (x_i)$ with pixel index i. Figure 1.3 illustrates several example images from different classes that arise in popular applications. Each of the images has n pixels and is therefore an element of $\mathbb{R}^n$.

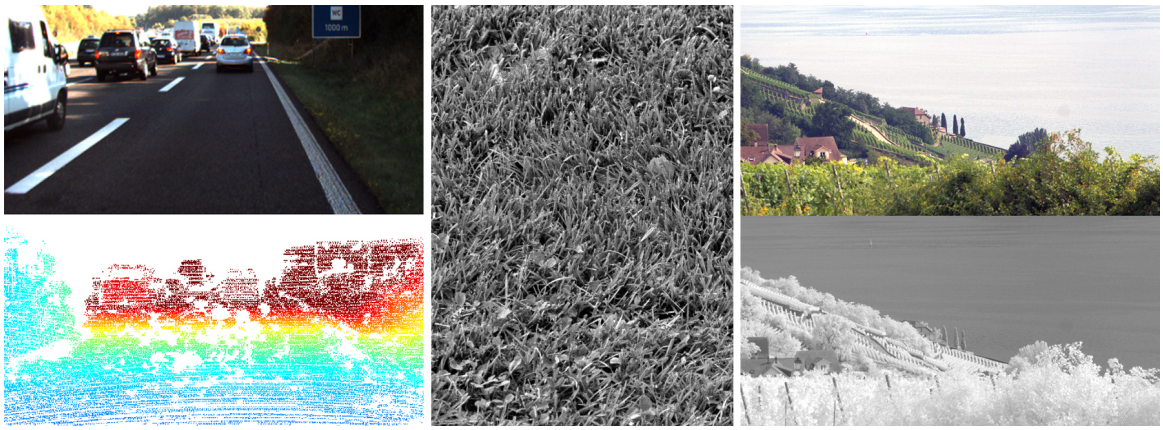


Figure 1.3: Digital image data acquired in different applications. *Left:* Color (top) and LiDaR image (bottom) from [169]. *Middle:* Gray-scale grass texture image from the Brodatz texture database. *Right:* Optical (top) and near infrared (bottom) photometric image from [25].

One can easily make the observation that the images that belong to a certain application do not occupy the entire space of all possible images but are rather scarce in their surrounding signal space $\mathbb{R}^n$. As a result, a common interpretation of data models is their description of signal space partitions that only contain the interesting image signals with respect to a certain application. Although the pixel intensity tensor closely resembles the structure of a discretized physical image, it is not the most efficient representation from an information-theoretic point of view [34]. In other words, it is difficult to find a simple description of the interesting signal space partitions in this pixel intensity domain. To address this, methods exist that transform the image data from this domain to another where the formulation of model fitness is more efficient. To that end, one can define a *transform* T on the class X of images which converts any image $\boldsymbol{x} \in \mathrm{X}$ to the transform space Z, yielding a new

representation $\boldsymbol{z}$. In short, this reads as

$$T\colon \mathrm{X} \to Z, \qquad \boldsymbol{x} \mapsto \boldsymbol{z} = T\boldsymbol{x}. \tag{1.7}$$

Furthermore, if any $\boldsymbol{x}$ can be restored without error from its representation $\boldsymbol{z}$, then the transform T is called *lossless* and there exists another transform $R\colon Z \to \mathrm{X}$ such that

$$\boldsymbol{x} = R\left(T\boldsymbol{x}\right), \qquad \forall \boldsymbol{x} \in \mathrm{X}.$$

The two transforms T and R are in the literature often referred to as *analysis* and *synthesis* transforms respectively [34]. The former owes its name to the fact that T transfers the signal $\boldsymbol{x}$ to a space where the analysis of its key features is straight-forward and the latter is to indicate that R composes an image in the pixel-intensity domain from its representation $\boldsymbol{z}$.

At first glance it seems intuitive that a model should make use of a lossless transform. As it turns out, however, many popular data models in the literature are lossy. Taking a closer look, a good model needs to focus on the key features of X by ignoring unimportant information and thus it is inevitably lossy [34, 173]. Central to establishing lossy models is the idea that the partitions of the signal space where images of interest reside, form subspaces of much lower dimension than the surrounding signal space. Approaches that follow this principle to model image data are abundant in the image processing literature of the past decades [55]. The most popular ones among them include the Fourier, cosine and wavelet transforms [108, 110], Principal Component Analysis (PCA) and most recently sparsity-based transforms [12, 150] as well as different types of artificial neural networks [78, 173]. All of these methods have had tremendous success in signal and image processing applications. However, there is an important aspect that divides these methods into two groups: Fourier, cosine and wavelet transforms rely on analytical rules to reduce the dimensionality of the data, which are agnostic of the class of images they are applied to. They drive well-known image compression standards such as JPEG and JPEG2000 [112]. PCA, sparsity-based transforms and neural networks are adaptable. This means that their transforms are not fixed analytically but instead can be adjusted to the data they encode by tuning their parameters to training samples that are obtained from the image class of interest. Although this tuning can be resource demanding, the adaptability of such methods makes them particularly interesting

for modeling specific image classes and for many applications this has proven to reduce the overall model error [55].

Learning model parameters from examples

Models that allow their data structure describing rules to be adjusted to training data have sparked tremendous interest in recent years and the methods built on them have consistently been leading the score boards of many image processing solutions. The methods follow two conceptually different optimality paradigms to adjust their model parameters to training data: *Supervised learning* methods derive the optimality of their representations by explicitly comparing pairs of input and labeled output. Typically, the labeling of data is specific to a certain application and it is often costly to create large training sets that are required to learn the parameters of these models well. *Unsupervised learning* methods, on the other hand, solely rely on unlabeled examples to adjust the model to fit the input distribution. Without requiring labeled data, models tuned by such methods represent the input data to differentiate them by their inner structure and optimize their representation by application-independent optimality criteria. Classical examples of such methods are the k-means clustering algorithm [103] or PCA [140]. A recent and very successful unsupervised learning approach is that of sparsity based models. The key idea underlying these models is that informative image signals not only occupy a small volume of the signal space but also approximately reside in a union of low-dimensional subspaces. As a result, they admit a sparse representation, which means that they can be denoted by very few parameters with the help of an appropriate transformation [131]. This notion is related to well-known concepts in information theory such as the Minimum Description Length [16] or Kolmogorov complexity [95] and can even be traced back to the Middle Age principle of Occam's razor. The key to these representations, is to learn an optimal transform under the sparsity principle for a given set of training data from the class of interest. In accordance with the analysis and synthesis transforms introduced earlier, two different approaches on constructing such sparsity promoting transforms have emerged in the literature.

Sparsity-based models for image data

The most popular sparse signal model in the literature is the *sparse synthesis model*. It is a generative model and assumes that a signal $\boldsymbol{x} \in \mathbb{R}^n$ can be constructed from a linear combination of a small number s of elements $\boldsymbol{d}_i$ from a collection of prototype signals $\{\boldsymbol{d}_i\}_{i=1}^{l}$. The prototype images are collected as columns of a matrix $\boldsymbol{D} \in \mathbb{R}^{n \times l}$ which is referred to as the dictionary. The coefficient vector containing the weights for the linear combination is denoted as $\boldsymbol{z} = [z_1, \ldots, z_l]^\top$. An image can then be generated under this model by the matrix vector multiplication

$$\boldsymbol{x} = \sum_{i=1}^{l} z_i \boldsymbol{d}_i = \boldsymbol{D}\boldsymbol{z}. \tag{1.8}$$

For regularizing inverse problems, one is interested in determining whether a given signal is represented by the model through a measure of model fitness. From Eq. (1.8) it can be observed that $\boldsymbol{D}$ generates samples of the class X as elements of the linear subspace spanned by some of its columns. These columns are indexed by the *support* of $\boldsymbol{z}$ which is assumed to contain only few elements. The support of a vector is defined as the set of its indices whose coefficients are different from zero.

$$\operatorname{supp}(\boldsymbol{z}) := \{\, i \,|\, z_i \neq 0 \,\}. \tag{1.9}$$

To verify if an image $\boldsymbol{x} \in \mathbb{R}^n$ was generated by $\boldsymbol{D}$ and therefore belongs to X, one needs to confirm that it lies in a subspace of $\mathbb{R}^n$ that is spanned by few of the columns of $\boldsymbol{D}$. For this purpose, its representation $\boldsymbol{z}$ needs to be found and subsequently confirmed that the cardinality of its support $|\operatorname{supp}(\boldsymbol{z})|$ is small. In the case where $\boldsymbol{D}$ is a basis of $\mathbb{R}^n$, the representation is uniquely defined as $\boldsymbol{z} = \boldsymbol{D}^{-1}\boldsymbol{x}$. To measure the model fitness, the support cardinality of the representation $\boldsymbol{z}$ is evaluated, denoted by the ℓ_0-pseudo-norm

$$\|\boldsymbol{z}\|_0 := |\operatorname{supp}(\boldsymbol{z})|. \tag{1.10}$$

However, if the number of atoms in the dictionary is raised above l to increase the descriptiveness of the model, recovering optimally sparse representations becomes challenging [50]. Known as the *sparse coding problem*, the more general goal of finding the sparsest

representation of a signal over a given dictionary is formalized as

$$\min \|\boldsymbol{z}\|_0 \quad \text{subject to} \quad \boldsymbol{x} = \boldsymbol{D}\boldsymbol{z}. \tag{1.11}$$

To use this model as a regularizer in inverse problems, Eq. (1.11) needs to be solved. Despite the dictionary being a linear transform, finding an exact solution is NP-complete [45] due to the ℓ_0-term. Solving this problem approximately has however been studied extensively [54].

More recently, the analysis transform perspective to sparse representations has sparked much research interest. Often coined as the sparse synthesis model's fraternal twin, it is aimed at addressing the transformation of a signal to a sparse representation more directly and has been popularized under the term *co-sparse analysis model* [53, 123]. In this model, an *analysis operator* $\boldsymbol{\Omega} \in \mathbb{R}^{l \times n}$ is defined such that the representation vector $\boldsymbol{z} = \boldsymbol{\Omega}\boldsymbol{x}$ is sparse, i.e. $\|\boldsymbol{\Omega}\boldsymbol{x}\|_0$ is small. In sparsity-based models, one usually deals with *overcomplete* transforms, which implies $l > n$ and in this case, a direct connection between the synthesis and the analysis model through the inverse $\boldsymbol{\Omega} = \boldsymbol{D}^{-1}$ does not hold [53, 123]. Geometrically interpreted, the signal encoded by the analysis model resides in the intersection of hyperplanes defined by the normals that are the rows of the analysis operator $\boldsymbol{\Omega}$ and whose corresponding entries in $\boldsymbol{z}$ are *equal to zero*. This is in contrast to the synthesis model, where the *non-zero* coefficients of the representation determine the columns of $\boldsymbol{D}$ that span the signal space. To emphasize this difference, the number of entries that are equal to zero in the co-sparse analysis model is referred to as the co-sparsity p of the model.

So far, it was assumed that the transform operators are given and static. In fact, several methods that occur ubiquitously in signal and image processing solutions can be interpreted as such transforms. Prominent examples are Gabor filters, wavelets or the discrete version of Total Variation (TV) prior [151]. What makes these models even more interesting is that the coefficients of $\boldsymbol{\Omega}$ can be tuned automatically to maximize co-sparsity over a given set of training images and therefore adapt the transform to best represent a specific image class [149]. The parameter learning is a challenging problem in itself, which is discussed in detail in Chapter 2.

Considering again the inverse problem in Eq. (1.6), a measure of fitness under the analysis model needs to be defined. Given that images of the class X are described well if $\boldsymbol{\Omega}\boldsymbol{x}$ contains

many entries close to zero, an obvious choice for measuring model fitness is to measure the sparsity of $\boldsymbol{\Omega x}^\star$ for a candidate $\boldsymbol{x}^\star$. Accordingly, the reconstruction problem in Eq. (1.6) becomes

$$\boldsymbol{x}^\star \in \operatorname{argmin} \|\boldsymbol{\Omega x}\|_0 \quad \text{subject to} \quad f\left(\boldsymbol{y}, \boldsymbol{A x}\right) \leq \varepsilon. \tag{1.12}$$

Solving this problem again has combinatorial complexity, but as it will turn out in the next chapter, good approximate solvers exist.

Model-based learning of co-sparse image representations

In all previous considerations, the system generating the image signals that make up the class of interest has been treated as a black box and the modeling of this data is purely based on the distributions of its observations. However, in many image processing and computer vision applications well-justified analytic information of respective data generators is available. This work is motivated by the idea of constructing models of image data that are based on the distribution of observations and are consistent with application-specific knowledge. To that end, co-sparse analysis models are combined with analytic insights of multiple prominent image processing applications and novel methods are developed for analysis operator learning that still rely on unlabeled training data but incorporate application-specific terms in their learning objectives. This is referred to as *model-based learning of co-sparse representations*.

1.2 Research goals

With the ever-increasing volume of recorded digital image data and the raising demand to automatically analyze and interpret it, the quest to find better models for various types of image data remains an active research frontier. Over the past few years, much effort has been invested in developing efficient methods to learn sparse data models from data samples. The co-sparse analysis model has been of particular interest recently, both for investigation of its theoretical properties as well as its usefulness in applications. In this thesis, it is investigated how the learning of co-sparse image representations can be improved with model assumptions of particular image processing applications. Its goal is to improve the optimization objective

to achieve better representations that generate more accurate application results and require fewer training samples to learn model parameters reliably. Specifically, the following aspects are addressed:

Centered analysis operator learning

Undeniably, pre-processing of raw data is an important step for any data investigation. It often incorporates specific knowledge with the goal of normalizing the data while preserving relevant information for further analysis. Normalizing the mean and variance of input data are arguably among the most common methods in that regard and well justified from the perspective of photometric image formation with bias and gain. Although being a common practice for learning parameters of co-sparse analysis models, its impact on the geometry of the data and therefore the learning outcome is mostly overlooked in the literature. It motivates to investigate the illumination normalization model within analysis operator learning and to find a way to exploit its geometric property within an efficient optimization framework.

Co-sparse analysis model for multi-modal image data

Despite the advent of multi-image-sensor devices and setups in many engineering fields, existing sparsity models focus mostly on unimodal image data. The different properties that are encoded in multi-modal sensor data collected from the same environment often lead to related patterns in the data due to common underlying physical phenomena. It suggests that a suitable signal representation of multi-modal image data should reflect such interdependencies and make them easily accessible. So far, pioneering work on multi-modal image models based on joint sparsity has only considered the synthesis perspective. It motivates to explore image models based on jointly co-sparse representations and to develop appropriate algorithms which allow learning model parameters automatically from training data and demonstrate their effectiveness in real applications that rely on bimodal image data.

Furthermore, the integration of image data from multiple sensors is challenging in practice

due to differing resolution and noise characteristics of the sensors as well as misalignment of the data. It is desirable to develop methods that address these difficulties by leveraging the exposed interdependent patterns captured in a suitable multi-modal signal model. Specifically, a jointly co-sparse analysis model of bimodal image data should be investigated for usage as a regularizer in solving inverse problems such as super-resolution, denoising, in-painting and image registration.

Co-sparse representations with spatial regularity

In image segmentation tasks, data term and segment priors are decisive factors for the quality of the image partitioning. Comparing the structural similarity of local neighborhoods in the data term has proven more robust than comparing pixel-values. Designing representations that exhibit the relevant structural properties well, however, is time-consuming and data-dependent. Recent attempts at learning such representations from example images have shown to be a promising direction and motivate to investigate the learning of co-sparse representations based on a segmentation model. Narrowing the scope to the application of unsupervised texture segmentation, it shall be studied how a learned analysis operator can be employed as structural feature extractor. Guided by the fact, that texture segments are piece-wise constant in the image plane, it motivates to incorporate the property of spatial regularity into the signal model to learn better representations of the segments in a single textured image and use them in an existing segmentation framework to improve unsupervised texture segmentation.

1.3 Contributions

In the course of this work, several novel models and algorithms based on the co-sparse analysis framework were developed and are presented for the purpose of obtaining better representations of structure in image data and improved results in image processing applications.

In the first part, the co-sparse analysis model with centered operators is proposed for

brightness-normalized image data. Based on the observation that zero-mean training data introduces trivial solutions in existing analysis operator learning approaches, a method is developed to learn analysis operators that considers the geometric structure imposed by the pre-processing of the data and which avoids the trivial solutions. This is achieved by restricting solutions of the analysis operator learning task to an appropriately chosen manifold structure. Instead of learning analysis operators as an element of the oblique manifold, it is proposed to optimize over the intersection of a product of unit spheres and the zero-mean plane which forms a smooth manifold. By appropriately adapting an existing geometric conjugate gradients algorithm, it is shown how analysis operators for this model can be learned from real data efficiently. Subsequent experiments on a practical image reconstruction problem show that representations obtained through this model lead to better quality and require fewer training samples than the previous approach.

In the second part, a bimodal co-sparse analysis model is introduced that is able to capture the interdependence of local structure in two image modalities. It is based on the assumption that if a scene is captured by different sensor devices, e.g. intensity and depth cameras, the inherent structures in the acquired signals are related. This structural relation is modeled by a pair of analysis operators which yield representations of the images with a large overlap in their co-supports. An algorithm is proposed to learn such analysis operator pairs from noiseless and spatially aligned training data. Furthermore, it is demonstrated how this model can be applied to regularize inverse problems and provide empiric results on different data sets demonstrating its effectiveness in image-guided depth map reconstruction. In addition, a method is developed that employs the joint bimodal co-sparse model as a prior for rigid image registration. A new algorithm is provided which allows to spatially register intensity-depth and intensity-NIR image pairs, that are misaligned under different types of rigid transformations.

In the third part, a novel method for unsupervised segmentation of color texture mosaics is proposed. Its main contributions are a model for learning representations from non-annotated texture images that capture the inner structure of local textures and a practical algorithm that automatically segments a texture image based on the learned representations. The model is based on the observation that in existing segmentation methods, image descriptors are often designed such that the feature image is approximately constant on a

texture segment. The basic idea of the proposed model is to learn an analysis operator that yields approximately piecewise constant supports of local neighborhoods on the pixel grid. Besides the constraints used in the first part for learning co-sparse representations, the learning objective is to minimize the cost function of the popular piecewise constant Mumford-Shah segmentation model, i.e. the total length of the discontinuity set of the corresponding feature image. Furthermore, a segmentation algorithm is developed based on the existing Lagrange formulation of the piecewise constant Mumford-Shah model. As the data term, a Mahalanobis distance defined on the covariances of the support elements obtained by the learned analysis operator is considered. It is demonstrated empirically that the method achieves state-of-the-art results in evaluations on standard datasets for unsupervised texture segmentation and that it has great potential in being effective in segmenting histology images.

1.4 Thesis outline

Having set the key points that motivate model-based learning of co-sparse image representations in this initial chapter, the remainder of this thesis is organized as follows:
Chapter 2 provides a more thorough introduction of the co-sparse analysis model and summarizes related prior art on analysis operator learning algorithms. The subsequent chapters discuss the main contributions of this thesis in detail and are based on the peer-reviewed publications indicated in the opening paragraph of each respective chapter. In Chapter 3, the centered co-sparse analysis model is presented, along with numerical procedure for learning its parameters from training data. It further contains a short introduction to geometric gradient methods on matrix manifolds which are used in all presented numerical algorithms. The joint co-sparse analysis model for bimodal image data is presented in Chapter 4. It contains the description of the learning algorithm and introduces how inverse problems for bimodal image reconstruction tasks can be effectively regularized by this new model. Empirical studies on different datasets demonstrate its effectiveness on image-guided depth-map reconstruction. In addition, it describes how the model is further useful as a prior in bimodal image alignment, which is experimentally validated and compared with classic approaches. Learning representations for unsupervised texture segmentation is the focus of Chapter 5.

It presents a model of local image structure with spatial regularity within the framework of geometric analysis operator learning. It also contains an empirical study of the proposed segmentation algorithm including a comparison with several state-of-the-art texture segmentation algorithms and extensive parameter sensitivity analyses. Finally, Chapter 6 describes the software implementation and its architecture which was designed to conduct all numerical experiments throughout this thesis, before closing this work with concluding remarks in Chapter 7.

Chapter 2

Prior art on learning co-sparse image representations

The methods developed in this work for modeling image data for the applications described in subsequent chapters are based on the co-sparse analysis model [53]. In this chapter, the foundations of this model and in particular the problem of learning its parameters from data are discussed in detail along with a review of the most relevant prior work from the recent literature.

2.1 Co-sparse analysis model for noisy data

Recollect from the introduction in the previous chapter that the analysis model relies on a linear operator $\boldsymbol{\Omega} \in \mathbb{R}^{l \times n}$ that is typically over-complete, i.e. $l > n$. When applied to a signal $\boldsymbol{x} \in \mathbb{R}^n$, it yields a representation $\boldsymbol{z} = \boldsymbol{\Omega}\boldsymbol{x}$ that is sparse. The operator $\boldsymbol{\Omega}$ is known as the *analysis operator* and the number of vanishing coefficients in the representation $\boldsymbol{z} \in \mathbb{R}^l$ is referred to as *co-sparsity* [148]. It emphasizes the fact that the zero-components in the representation determine the subspace in which the signal is located and it is defined as

$$p = l - \|\boldsymbol{z}\|_0. \tag{2.1}$$

Concretely, a signal $\boldsymbol{x}$ that is p-cosparse, i.e. p coefficients in $\boldsymbol{z} = \boldsymbol{\Omega}\boldsymbol{x}$ are zero, is orthogonal

to p rows of $\boldsymbol{\Omega}$ and resides in the orthogonal complement of the space spanned by these rows. From a geometrical perspective, the class $\mathrm{X} \subseteq \mathbb{R}^n$ of signals that are modeled by $\boldsymbol{\Omega}$ is contained in a union of subspaces [101] and one such $\boldsymbol{x} \in \mathrm{X}$ lies in the intersection of all hyperplanes whose normal vectors are given by the rows of $\boldsymbol{\Omega}$ that are indexed by the zero entries of $\boldsymbol{\Omega x}$. This index set is called the *co-support* of $\boldsymbol{x}$ and is denoted by

$$\operatorname{cosupp}(\boldsymbol{\Omega x}) := \{j \mid (\boldsymbol{\Omega x})_j = 0\}, \tag{2.2}$$

where $(\boldsymbol{\Omega x})_j$ is the j-th entry of the analyzed vector $\boldsymbol{z}$.

In contrast to the synthesis model where the number of non-zero components of $\boldsymbol{z}$ in Eq. (1.8) can become arbitrarily small, the analysis model leads to milder sparsity $\|\boldsymbol{\Omega x}\|_0 \geq l - n$, since otherwise n or more rows of $\boldsymbol{\Omega}$ would need to be orthogonal to the signal. Assuming the operator is in general position, however, this is not possible if neither $\boldsymbol{x}$ nor rows of $\boldsymbol{\Omega}$ become zero and consequently, $0 \leq p \leq n$ [133, 150].

2.1.1 Analysis pursuit

One convenience of the analysis model is that the representation of an arbitrary signal $\boldsymbol{x}$ can be obtained readily by its multiplication with $\boldsymbol{\Omega}$ and one easily determines model fitness by measuring its co-support cardinality. In reality however, samples that are obtained in real applications will likely not be available directly, but only through some noisy linear measurements $\boldsymbol{y} = \boldsymbol{Ax} + \boldsymbol{n}$, where $\boldsymbol{n}$ is some bounded noise and $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is a measurement operator [29]. As a result, the analyzed version of the noisy observation is only approximately co-sparse and the recovery of a p-cosparse signal $\boldsymbol{x}$ from $\boldsymbol{y}$ is challenging, even if $\boldsymbol{A}$ is the identity. Consequently, before discussing the task of adapting analysis operators to data, this problem, referred to as *analysis sparse coding* or *analysis pursuit*, needs to be considered. Regarding the sparsity of the analyzed signal as the measure of model fitness and assuming Gaussian noise, we can rewrite the inverse problem from Eq. (1.6) as one of the two forms

$$\boldsymbol{x}^\star = \operatorname*{argmin}_{\boldsymbol{x}} \|\boldsymbol{\Omega x}\|_0 \quad \text{subject to} \quad \|\boldsymbol{Ay} - \boldsymbol{x}\|_2^2 \leq \varepsilon \tag{2.3}$$

$$\boldsymbol{x}^\star = \operatorname*{argmin}_{\boldsymbol{x}} \|\boldsymbol{Ay} - \boldsymbol{x}\|_2^2 \quad \text{subject to} \quad \|\boldsymbol{\Omega x}\|_0 = l - p, \tag{2.4}$$

and they are equivalent if a correct correspondence between ε and p is be established [148].

The analysis pursuit problem is combinatorial in nature and therefore computationally intractable in practice [148]. However, a number of methods have been developed to find good approximate solutions and they can be grouped in three categories depending on their approximation strategies: relaxation of the ℓ_0-term in Eq. (2.3) or Eq. (2.4) by its convex ℓ_1-surrogate, greedy heuristics to find a locally optimal choice of representation coefficients and hybrid greedy methods.

Analysis ℓ_1-minimization

The earliest and best-known way to deal with the computationally intractable ℓ_0-term in the Eq. (2.3) is to replace the pseudo-norm with a convex surrogate function. Using the ℓ_1-norm instead has proven very effective in making the problem numerically accessible and still promoting sparsity of the solution. In fact under certain conditions on $\boldsymbol{\Omega}$, analysis ℓ_1-minimization leads to the same solution as the original NP-hard problem [50, 30]. The resulting ℓ_1-analysis minimization problem accordingly reads as:

$$\boldsymbol{x}^\star = \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{\Omega}\boldsymbol{x}\|_1 \quad \text{subject to} \quad \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \leq \varepsilon. \tag{2.5}$$

Since Eq. (2.5) is convex, very efficient solvers exist [22, 23] to compute a solution numerically along with recovery and convergence guarantees, see e.g. [53, 156, 26, 123, 29, 172].

Greedy methods

Inspired by methods that make use of the structure of the sparse coding problem in the synthesis model, greedy algorithms that iteratively find locally optimal solutions to the analysis pursuit problem were developed.

Greedy Analysis Pursuit (GAP), proposed by Nam et al. in [123, 121], is an adaptation of the Orthogonal Matching Pursuit (OMP) [109, 139] for the analysis model. Let Λ denote the sought co-support with a target of p entries and let $\boldsymbol{\Omega}_\Lambda$ correspond to the analysis operator

with only the rows that are indexed by the co-support Λ. Starting from a full co-support $\hat{\Lambda}_0 = \{1, \ldots, l\}$, GAP then aims to reduce the index set to size p. The initial estimate of the recovered signal $\hat{\boldsymbol{x}}_0$ is set to

$$\hat{\boldsymbol{x}}_k = \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{\Omega}_{\hat{\Lambda}_k} \boldsymbol{x}\|_2^2 \quad \text{subject to} \quad \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}, \tag{2.6}$$

with $k = 0$. The linear operator $\boldsymbol{A}$ is used in the original work [123] but for simplicity, here it is assumed that it is the identity $\boldsymbol{A} = \boldsymbol{I}$. At each iteration k of GAP, the current estimate $\hat{\boldsymbol{x}}_{k-1}$ is multiplied with $\boldsymbol{\Omega}$. The index of the analyzed vector $\boldsymbol{z} = \boldsymbol{\Omega}\hat{\boldsymbol{x}}_{k-1}$ whose coefficient is the largest, gets removed from the co-support, i.e. $\hat{\Lambda}_k = \hat{\Lambda}_{k-1} \setminus \left\{\operatorname{argmax}_{i \in \hat{\Lambda}_{k-1}} |z_i|\right\}$. Finally the estimate of the recovered signal $\hat{\boldsymbol{x}}_k$ is updated using Eq. (2.6). The algorithm stops either after a fixed number of iterations determined by $l - p$ or if the difference between two successive estimates is small, which constitute standard stopping criteria for such iterative optimization procedures. Note that this description of GAP deviates slightly from the original version with respect to the chosen measurement operator. In the original work, a more general linear operator $\boldsymbol{A}$ is admitted and the recovery can still succeed under certain conditions which are discussed in [123] and [121]. Formally, the only difference is that instead of Eq. (2.6), the signal estimate is updated using

$$\hat{\boldsymbol{x}}_k = \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \quad \text{subject to} \quad \|\boldsymbol{\Omega}_{\hat{\Lambda}_k} \boldsymbol{x}\|_2^2 = 0. \tag{2.7}$$

It is typically solved for high-dimensional data in its unconstrained form

$$\hat{\boldsymbol{x}}_k = \underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda \|\boldsymbol{\Omega}_{\hat{\Lambda}_k} \boldsymbol{x}\|_2^2, \tag{2.8}$$

with a small weight λ. To summarize, GAP iteratively seeks the non-zero coefficients of the representation by sequentially selecting the indexes of operator rows that are least correlated with the measurements and removing them from the co-support.

The opposite view, finding the zero-components of the representation, is taken in the approach of Rubinstein et al. which culminates in the Backward Greedy (BG) algorithm described in [148, 150]. Here, the initial estimate of the recovered signal is set to $\hat{\boldsymbol{x}}_0 = \boldsymbol{y}$ and the co-support starts empty $\hat{\Lambda}_0 = \emptyset$. Identically with GAP, in each iteration k of BG the analyzed version of the previous signal estimate $\boldsymbol{z} = \boldsymbol{\Omega}\hat{\boldsymbol{x}}_{k-1}$ is considered. However, instead

of discarding the index of the most correlated row in $\hat{\Lambda}_k$, BG adds the index of the row that is least correlated with $\hat{\boldsymbol{x}}_{k-1}$ to the co-support, i.e. $\hat{\Lambda}_k = \hat{\Lambda}_{k-1} \cup \left\{\text{argmin}_{i \notin \hat{\Lambda}_{k-1}} |z_i|\right\}$. One iteration of BG is finalized by updating the signal estimate $\hat{\boldsymbol{x}}_k$ by projecting the measurements on the orthogonal complement of the rows of $\boldsymbol{\Omega}$ indexed by the current co-support estimate. Denoting the pseudo-inverse with †, this reads formally as

$$\hat{\boldsymbol{x}}_k = \left[\boldsymbol{I} - \boldsymbol{\Omega}_{\hat{\Lambda}_k}^{\dagger} \boldsymbol{\Omega}_{\hat{\Lambda}_k}\right] \boldsymbol{y}. \tag{2.9}$$

To solve the last step in each iteration efficiently, an orthogonalization scheme is proposed which circumvents the costly computation of the pseudo-inverse $\boldsymbol{\Omega}_{\hat{\Lambda}_k}^{\dagger}$ in Eq. (2.9). The algorithm terminates after p steps or if the difference between signal estimate and measurements $\|\hat{\boldsymbol{x}}_k - \boldsymbol{y}\|_2$ exceeds a threshold.

Rubinstein et al. also proposed an extension called Optimized Backward Greedy (OBG) algorithm. Instead of adding in each iteration the index of the analysis operator row to the co-support that yields the smallest correlation with the signal estimate, all possible $\hat{\Lambda}_k^{\text{temp}} = \hat{\Lambda}_{k-1} \cup \left\{i \notin \hat{\Lambda}_{k-1}\right\}$ are generated with their respective updates of the signal estimate $\hat{\boldsymbol{x}}_k^{\text{temp}}$. Finally, the co-support is extended by the index that leads to the smallest difference between the previous and the current estimate $\|\hat{\boldsymbol{x}}_k^{\text{temp}} - \hat{\boldsymbol{x}}_{k-1}\|_2$.

Greedy-like methods

Another class of algorithms that find approximate solutions to the analysis pursuit problem in Eq. (2.3) and Eq. (2.4) are referred to as greedy-like algorithms. They include Analysis Iterative Hard Thresholding (AIHT), Analysis Hard Thresholding Pursuit (AHTP), Analysis CoSaMP (ACoSaMP) and Analysis Subspace Pursuit (ASP) and were adapted to the analysis setting by Giryes et al. in [64, 63, 65] from respective methods for the synthesis sparse model Iterative Hard Thresholding (IHT) [20], Hard Thresholding Pursuit (HTP) [58], Compressive Sampling Matching Pursuit (CoSaMP) [125] and Subspace Pursuit (SP) [43]. All four of these algorithms iteratively estimate a p-cosparse version of linear measurements $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$. Analogously to the descriptions of GAP and BG above, the presentation here restricts itself to $\boldsymbol{A} = \boldsymbol{I}$.

In AIHT and AHTP, the estimate is initialized with $\hat{\boldsymbol{x}}_0 = \mathbf{0}$ and then in each iteration k, first $\hat{\boldsymbol{x}}_k^{\text{temp}}$ is updated temporarily with a gradient step of the data fidelity term to minimize $\|\boldsymbol{y} - \boldsymbol{A}\hat{\boldsymbol{x}}_{k-1}\|_2^2$ with a fixed or adaptive step size. Since this new estimate is not guaranteed to be p-cosparse, next, a projection of $\hat{\boldsymbol{x}}_k^{\text{temp}}$ onto a p-cosparse subspace is performed and the respective co-support $\hat{\Lambda}_k$ is recovered by finding a solution to

$$\hat{\Lambda}_k = \mathrm{S}_p(\hat{\boldsymbol{x}}_k^{\text{temp}}) := \operatorname*{argmin}_{\Lambda:|\Lambda|=p} \|\boldsymbol{\Omega}_\Lambda \hat{\boldsymbol{x}}_k^{\text{temp}}\|_2^2. \tag{2.10}$$

Several different strategies to implement the function S_p are proposed that find the zero-entries either based on thresholding of the smallest entries or by more sophisticated procedures that are tailored to specific analysis operators. The last step in each iteration of AIHT and AHTP is an update of the signal estimate based on the new co-support. In AIHT, this is simply achieved by orthogonally projecting the previous estimate onto the nullspace of the truncated analysis operator $\boldsymbol{\Omega}_\Lambda$, i.e.

$$\hat{\boldsymbol{x}}_k = \left[\boldsymbol{I} - \boldsymbol{\Omega}_{\hat{\Lambda}_k}^\dagger \boldsymbol{\Omega}_{\hat{\Lambda}_k}\right] \hat{\boldsymbol{x}}_k^{\text{temp}}. \tag{2.11}$$

In AHTP on the other hand, the estimate is updated in the last step of each iteration by finding a p-cosparse approximation with the best data fidelity, solving

$$\hat{\boldsymbol{x}}_k = \operatorname*{argmin}_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \quad \text{subject to} \quad \boldsymbol{\Omega}_{\hat{\Lambda}_k}\boldsymbol{x} = \mathbf{0}. \tag{2.12}$$

The algorithms are terminated if residual size or relative iteration change cross a predetermined threshold.

ACoSaMP and ASP take a different approach. They both initialize a residual $\boldsymbol{y}_0^r = \boldsymbol{y}$ and the co-support estimate $\hat{\Lambda}_0 = \{1, \ldots, l\}$. In each iteration k, the first step consists of finding new co-support elements Λ_Δ by selecting the $a \cdot p$ smallest coefficients of $\boldsymbol{\Omega}\boldsymbol{A}^{-1}\boldsymbol{y}_{k-1}^r$ [63] or by other strategies S_{ap} [65] and then updating a temporary co-support estimate with $\hat{\Lambda}_k^{\text{temp}} = \hat{\Lambda}_{k-1} \cap \Lambda_\Delta$. Here, ACoSaMP and ASP differ in the number of new elements they select for the co-support with $a = 1$ (ASP) and $a = \frac{2p-l}{p}$ (ACoSaMP). The second step of each iteration involves computing a new temporary estimate of the recovered signal based on the updated co-support. Analogously to the last step of AIHT/AHTP, one seeks $\hat{\boldsymbol{x}}_k^{\text{temp}} = \operatorname{argmin}_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2$ subject to $\boldsymbol{\Omega}_{\hat{\Lambda}_k^{\text{temp}}}\boldsymbol{x} = \mathbf{0}$. Subsequently, the co-support is

updated by finding $\hat{\Lambda}_k = \mathrm{S}_p(\hat{\boldsymbol{x}}_k^{\mathrm{temp}})$. In the next step, the signal estimate $\hat{\boldsymbol{x}}_k$ is updated. Equivalently to AIHT/AHTP, this step differs slightly for ACoSaMP and ASP. ACoSaMP updates the signal estimate by the orthogonal projection given in Eq. (2.11) while ASP updates with the most data consistent solution given in Eq. (2.12). Finally, the residual is updated $\boldsymbol{y}_k^r = \boldsymbol{y} - \boldsymbol{A}\hat{\boldsymbol{x}}_k$. In practice, the equality constraints in the update steps are relaxed to unconstrained minimization problems.

Having introduced the most important methods to recover co-sparse representations from arbitrary signals, the discussion of related work turns now to the actual learning of analysis operators from example image signals.

2.1.2 Analysis operator learning

In the discussion of the analysis pursuit problem in the previous section, it was assumed that the analysis operator $\boldsymbol{\Omega}$ is known and fixed. Indeed, many well known discrete transforms in image processing can be considered as analysis operators in the sense of the introduced framework. For example Gabor or Haar wavelets [111], curvelets [28], wave atoms [46], the fused Lasso [163] or finite differences operators can be considered as analytically crafted operators that yield sparse representations of image data [121]. To ease the task of hand-crafting an analysis operator that is most suitable for a specific class of image signals, *analysis operator learning* aims to automatically find an operator that best suits given example data.

To fit the co-sparse analysis model to a specific class of image signals at hand, one needs to tune the rows of $\boldsymbol{\Omega}$ such that the desired co-sparsity is achieved for samples that fall into the class of interest. To address this, let $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_M] \in \mathbb{R}^{n \times M}$ be a collection of M signals $\boldsymbol{y}_i \in \mathbb{R}^n$, which are considered to be noisy observations of signals $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_M]$ from a class X, e.g. $\boldsymbol{y}_i = \boldsymbol{x}_i + \boldsymbol{n},\ \boldsymbol{n} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$. The task of learning an analysis operator for X is then to find a suitable $\boldsymbol{\Omega}$ which provides the most co-sparse approximation $\boldsymbol{X}$ for the data matrix $\boldsymbol{Y}$ [183]. In other words, the representation matrix $\boldsymbol{Z} = \boldsymbol{\Omega}\boldsymbol{X}$ should contain as many small coefficients as possible. Formally, the learning objective to be minimized during

the learning of the operator from noisy samples can be written as

$$(\boldsymbol{\Omega}^\star, \boldsymbol{X}^\star) = \underset{\boldsymbol{\Omega},\boldsymbol{X}}{\operatorname{argmin}} \|\boldsymbol{\Omega}\boldsymbol{X}\|_0 \quad \text{subject to} \quad \|\boldsymbol{X} - \boldsymbol{Y}\|_F \leq \sigma. \tag{2.13}$$

The parameter σ represents an estimate of the expected noise power. Note that the notation is slightly abused in the sense that the ℓ_0-pseudo-norm of the matrix represents the sum of the norms of its columns, i.e. for some matrix $\boldsymbol{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r]$ it denotes

$$\|\boldsymbol{U}\|_0 = \sum_{i=1}^{r} \|\boldsymbol{u}_i\|_0. \tag{2.14}$$

The problem in Eq. (2.13) is highly non-convex and approximation techniques are required to find good local minimizers [150]. Although analysis operator learning has only been investigated recently, several successful methods have emerged and are briefly reviewed in the following.

Sequential Minimal Eigenvalues

One of the earliest works addressing the problem of analysis operator learning was published in [133]. There, Ophir et al. propose to learn the rows of $\boldsymbol{\Omega}$ sequentially. Starting from a randomly generated row $\hat{\boldsymbol{\omega}}$, its inner products with the training set $\hat{\boldsymbol{\omega}}^\top \boldsymbol{y}_j$ are computed. Then, first a threshold θ is set, such that $\frac{cpM}{l}$ of the inner products are below this threshold ($c \leq 1$). Second, the index of samples whose inner product is smaller than the threshold θ are recorded in the set $J = \{1 \leq i \leq M \mid \hat{\boldsymbol{\omega}}^\top \boldsymbol{y}_j < \theta\}$. Finally, the row $\boldsymbol{\omega}$ is set to the eigenvector which is associated with the smallest eigenvalue of $\boldsymbol{Y}_J \boldsymbol{Y}_J^\top$. These steps are repeated until the threshold θ is smaller than a predetermined value. To obtain all rows of $\boldsymbol{\Omega}$, this procedure is performed l times. In order to prevent duplicate rows, a different training set is randomly chosen from $\boldsymbol{Y}$ for every row. In addition, a newly obtained row is compared to existing ones and only added to $\boldsymbol{\Omega}$ if it is sufficiently different and otherwise another row is generated. One drawback of this algorithm is that the likelihood of finding a row that is similar to ones that are already in $\boldsymbol{\Omega}$ increases as the procedure progresses. This leads to many rejections of new rows and slows the procedure significantly. Also it becomes less efficient with growing signal dimension n [148].

Analysis K-SVD

The method proposed by Rubinstein et al. in [148] and [150] draws its spirit from the K-SVD algorithm [12] that solves the related dictionary learning problem in the synthesis model. The idea of *Analysis K-SVD* is to find a solution to Eq. (2.13) by alternatingly optimizing for the two target variables. In the first phase, a p-cosparse approximation of the signals in $\boldsymbol{Y}$ is optimized while keeping the analysis operator constant. In the second phase, the signal approximates are kept fixed and the analysis operator estimate is updated. The two phases are repeated until some stopping criterion is reached. It is assumed that all of the noiseless signals $\boldsymbol{x}_i$ are orthogonal to p rows of the final operator $\boldsymbol{\Omega} \in \mathbb{R}^{l \times n}$. The learning problem in Eq. (2.13) is slightly reformulated as

$$\begin{aligned} (\boldsymbol{\Omega}^\star, \boldsymbol{X}^\star) = \underset{\boldsymbol{\Omega}, \boldsymbol{X}}{\operatorname{argmin}} \quad & \|\boldsymbol{X} - \boldsymbol{Y}\|_F^2 \\ \text{subject to} \quad & \|\boldsymbol{\Omega}\boldsymbol{x}_i\|_0 \leq l - p, \quad \forall i \mid 1 \leq i \leq M \\ & \|\boldsymbol{\omega}_j\|_2 = 1, \qquad \forall j \mid 1 \leq j \leq l \end{aligned} \tag{2.15}$$

The normalization constraint, which forces the rows $\boldsymbol{\omega}_j$ of the operator (written as a column vector) to have unit norm, is introduced to prevent estimates to degenerate by scaling. During the first phase of Analysis K-SVD, finding the co-sparse approximations of each $\boldsymbol{y}_i$ is the analysis pursuit problem from Eq. (2.4) and solved using the BG algorithm or its variant OBG. Once the intermediate result $\hat{\boldsymbol{X}}$ is computed, the rows $\boldsymbol{\omega}_j$ are updated sequentially. Since the BG algorithm returns an estimate of the co-support $\hat{\Lambda}_i$ for each of the samples $\boldsymbol{y}_i$, one can easily determine the samples that are (approximately) orthogonal to $\boldsymbol{\omega}_j$ and collect their indexes in the set $J = \{1 \leq i \leq M \mid j \in \hat{\Lambda}_i\}$. Denoting this sample subset as the sub-matrix $\boldsymbol{Y}_J$ of $\boldsymbol{Y}$, the update step of $\boldsymbol{\omega}_j$ reads as

$$\hat{\boldsymbol{\omega}}_j = \underset{\boldsymbol{\omega}_j}{\operatorname{argmin}} \|\boldsymbol{\omega}^\top \boldsymbol{Y}_J\|_2^2 \quad \text{subject to} \quad \|\boldsymbol{\omega}_j\|_2 = 1. \tag{2.16}$$

The solution can be obtained efficiently by computing the Singular Value Decomposition (SVD) of $\boldsymbol{Y}_J$ and selecting the left singular vector that corresponds to the smallest singular value. A mechanism is used to resolve deadlock situations in iterations by discarding and randomly regenerating rows based on heuristics involving the number of associated samples

in $\boldsymbol{Y}_J$ and the maximal inner product between $\boldsymbol{\omega}_j$ and other rows, often referred to as *mutual coherence*. The authors see one advantage of their method in the decomposition of the sample matrix, which makes the update of individual rows independent from each other and hence they can be computed in parallel.

Constrained Analysis Operator Learning

A different approach on over-complete anlysis operator learning from noisy data is proposed by Yaghoobi et al. in [183, 184, 185], coined Constrained Analysis Operator Learning (CAOL). They replace the ℓ_0-term in Eq. (2.13) with its ℓ_1-surrogate and observe that the learning problem naturally includes trivial and useless solutions, that need to be avoided by enforcing additional constraints $\mathcal{C}$ on estimates of $\boldsymbol{\Omega}$. Analogously to Eq. (2.14), $\|\boldsymbol{U}\|_1$ is used short for $\sum_i^r \|\boldsymbol{u}_i\|_1$. The problem is then formulated as

$$\begin{aligned} (\boldsymbol{\Omega}^\star, \boldsymbol{X}^\star) = \underset{\boldsymbol{\Omega}, \boldsymbol{X}}{\operatorname{argmin}} \quad & \|\boldsymbol{\Omega}\boldsymbol{X}\|_1 \\ \text{subject to} \quad & \|\boldsymbol{X} - \boldsymbol{Y}\|_F \leq \sigma, \\ & \boldsymbol{\Omega} \in \mathcal{C}_{\text{CAOL}}. \end{aligned} \tag{2.17}$$

The authors develop the constraint set $\mathcal{C}_{\text{CAOL}}$ from the following observations. First, the minimizer $\boldsymbol{\Omega} = \boldsymbol{0}$ of the cost function in Eq. (2.17) is obviously undesirable, as it contains no information, and a common way to avoid estimates to shrink to zero is to fix their norm. Evidently, this can be achieved by fixing the row norm, e.g. $\|\boldsymbol{\omega}_j\|_2 = 1$ for the j-th row of $\boldsymbol{\Omega}$, which is also employed by Analysis K-SVD in Eq. (2.16). However, it was found that row norm constraints are insufficient, since a single row which minimizes $\|\boldsymbol{\omega}^\top \boldsymbol{Y}\|_1$ could be repeated $l - 1$ times to construct the minimizer $\boldsymbol{\Omega}$ of Eq. (2.17). Since the rank of such a solution would equal one, the norm constraint alone will not help to find a suitable analysis operator. Further, the authors find that full-rank, i.e. $\operatorname{rank}(\boldsymbol{\Omega} = n)$, and tight-frame ($\boldsymbol{\Omega}^\top \boldsymbol{\Omega} = \boldsymbol{I}$) constraints avoid rank deficiency and ill-conditioning respectively, but are not sufficient for over-complete operators with $\boldsymbol{\Omega} \in \mathbb{R}^{l \times n}$, $l > n$. As a solution, Yaghoobi et al. propose to combine the row norm with the tight frame constraint, resulting in a Uniform

Normalized Tight Frame (UNTF) defined by

$$\mathcal{C}_{\text{CAOL}} = \left\{ \boldsymbol{\Omega} \in \mathbb{R}^{l \times n} : \boldsymbol{\Omega}^\top \boldsymbol{\Omega} = \boldsymbol{I},\ \forall 1 \leq j \leq l\ \|\boldsymbol{\omega}_j\|_2 = 1 \right\}. \tag{2.18}$$

In addition, the sparsity and data fidelity terms in learning problem Eq. (2.17) are reformulated as an unconstrained optimization problem using a Lagrangian multiplier λ, which leads to the CAOL learning problem

$$\begin{aligned} (\boldsymbol{\Omega}^\star, \boldsymbol{X}^\star) = \underset{\boldsymbol{\Omega}, \boldsymbol{X}}{\operatorname{argmin}} \quad & \|\boldsymbol{\Omega}\boldsymbol{X}\|_1 + \frac{\lambda}{2}\|\boldsymbol{X} - \boldsymbol{Y}\|_F^2 \\ \text{subject to} \quad & \boldsymbol{\Omega} \in \mathcal{C}_{\text{CAOL}} \end{aligned} \tag{2.19}$$

The UNTF constraint is not convex but Yaghoobi et al. propose a variational technique to find local optima. More precisely, in their Analysis Operator Learning Algorithm (AOLA) [185], the two target variables $\boldsymbol{\Omega}$ and $\boldsymbol{X}$ are updated in an alternating fashion by keeping one fixed while updating the other, minimizing Eq. (2.19). In each iteration k, *step 1* updates the analysis operator according to $\hat{\boldsymbol{\Omega}}_k = \operatorname{argmin}_{\boldsymbol{\Omega} \in \mathcal{C}} \|\boldsymbol{\Omega}\hat{\boldsymbol{X}}_k\|_1$. This non-convex sub-problem is solved by taking a negative step along its gradient $\nabla_{\boldsymbol{\Omega}}$ in the ambient space of the constraint set and subsequently projecting onto the UNTF. A subgradient is chosen randomly at the origin, where $\|\cdot\|_1$ is not differentiable, and the step size η is determined by a backtracking line-search along the gradient direction. The projection of the estimate after the gradient step is carried out again in two steps, respectively projecting onto the uniform normalized (UN) and the tight frame (TF) constraint sets. The projection $\mathcal{P}_{UN}$ is accomplished by scaling each row of $\boldsymbol{\Omega}$ to unit length or generating a random unit norm row if $\|\boldsymbol{\omega}\|_2 = 0$. The projection $\mathcal{P}_{TF}$ of a full-rank matrix onto the tight frame is carried out using the singular value decomposition $\boldsymbol{\Omega} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$ and replacing the diagonal matrix $\boldsymbol{\Sigma}$ with the identity matrix $\boldsymbol{I}$, resulting in $\mathcal{P}_{TF}(\boldsymbol{\Omega}) = \boldsymbol{U}\boldsymbol{I}\boldsymbol{V}^\top$. Finally, the analysis operator update in iteration k is obtained by $\hat{\boldsymbol{\Omega}}_k = \mathcal{P}_{UN}(\mathcal{P}_{TF}(\hat{\boldsymbol{\Omega}}_{k-1} - \eta\nabla_{\boldsymbol{\Omega}}))$. Although theoretically, alternating the projections $\mathcal{P}_{UN}$ and $\mathcal{P}_{TF}$ does not guarantee to converge to a UNTF, the authors report that even a single pair of projections are sufficient in practice. *Step 2* of an AOLA iteration considers the update of the co-sparse approximations $\hat{\boldsymbol{X}}$ of the data matrix. It consists of solving Eq. (2.19) for $\boldsymbol{X}$ using the newly updated $\hat{\boldsymbol{\Omega}}_k$ and is equal to analysis ℓ_1-minimization as discussed in the previous section. The Alternating Direction Method of Multipliers (ADMM) [52] is used to compute a solution to the convex program.

Steps 1 and 2 are alternated until a standard stopping criterion (see above) is reached and the algorithm returns the final estimates as the sought solution $(\boldsymbol{\Omega}^\star, \boldsymbol{X}^\star)$. Initially, CAOL was proposed for the noiseless case [183], i.e. $\boldsymbol{Y} = \boldsymbol{X}$, and extended to the noisy case in [184]. The Lagrangian multiplier λ in Eq. (2.19) determines how much noise is allowed in the model, becoming the noiseless case in the limit $\lambda \to \infty$. Empirically though, it is shown that modeling the noise leads to much higher co-sparsity in the learned representations [185].

Geometric Analysis Operator Learning

The method of Hawe et al., called GeOmetric Analysis operator Learning (GOAL), also addresses the noisy analysis model. Unlike Analysis K-SVD and CAOL, which require the target co-sparsity p as a parameter to their algorithms to find approximations that are exactly p-cosparse, the idea of GOAL is to minimize empirical mean and variance of the sparsity in $\boldsymbol{\Omega Y}$ directly. Although Analysis K-SVD and CAOL are designed for the same objective, GOAL allows some signals to deviate in the co-sparsity of their representations and does not explicitly approximate a co-sparse data matrix $\boldsymbol{X}$. The authors argue that for signal classes as diverse as natural images, this is a more realistic setup than requiring all signals to lie on the same dimensional subspace. Furthermore, they use the non-convex ℓ_p-surrogate function for ℓ_0 with $0 < p \leq 1$, which has been shown to perform well for sparse recovery in the synthesis model [35]. It is commonly defined as

$$\|\boldsymbol{u}\|_p^p = \sum_i |u_i|^p. \tag{2.20}$$

The objective of minimizing the sum of the empirical variance and the squared empirical mean of the sample sparsity in combination with this non-convex sparsity measure, renders the GOAL problem as

$$\begin{aligned} \boldsymbol{\Omega}^\star = \underset{\boldsymbol{\Omega}}{\operatorname{argmin}} \quad & \tfrac{1}{2M} \sum_{i=1}^{M} \left(\tfrac{1}{p} \|\boldsymbol{\Omega} \boldsymbol{y}_i\|_p^p \right)^2 \\ \text{subject to} \quad & \boldsymbol{\Omega} \in \mathcal{C}_{\text{GOAL}} \end{aligned} \tag{2.21}$$

It is important to note that the data fidelity term $\|\boldsymbol{X} - \boldsymbol{Y}\|_F$ is omitted completely which

relates to the CAOL problem for $\lambda \to 0$. This means that the weak supervision by the denoising task is removed, and instead useless minimizers of the cost function need to be prevented by appropriate constraints $\mathcal{C}_{\mathrm{GOAL}}$. To that end, Hawe et al. follow a similar path as [185] in that they aim to prevent solutions that are scale-degenerate, rank-deficient or contain duplicate rows.

Analogously to Analysis K-SVD and CAOL, GOAL restricts operator rows $\boldsymbol{\omega}_j$ to unit Euclidean norm. Furthermore, the authors recognize that the norm constraint imposes a geometric structure on the problem, which is known in its transposed form as the product manifold of unit spheres or *oblique manifold*. It is defined in [11, 10] as

$$\mathrm{OB}(n,l) := \{\boldsymbol{W} \in \mathbb{R}^{n\times l} : (\boldsymbol{W}^\top \boldsymbol{W})_{ii} = 1,\ 1 \leq i \leq l\}. \tag{2.22}$$

In other words, the sought analysis operator is restricted to $\boldsymbol{\Omega}^\top \in \mathrm{OB}(n,l)$.

In contrast to CAOL, rank-deficiency and conditioning of $\boldsymbol{\Omega}$ are not controlled by restricting iterates to a tight frame. Instead, two log-barrier functions are introduced to the learning objective that serve as soft-constraints on mutual coherence and the condition of $\boldsymbol{\Omega}$. Since the Gramian matrix $\boldsymbol{\Omega}^\top\boldsymbol{\Omega}$ of a full rank analysis operator ($\mathrm{rank}(\boldsymbol{\Omega}) = n$ for $l \geq n$) is positive definite, its determinant is strictly positive, i.e. $\det(\boldsymbol{\Omega}^\top\boldsymbol{\Omega}) > 0$. Consequently, Hawe et al. propose to prevent rank-deficiency of $\boldsymbol{\Omega}$ by penalizing iterates whose Gramian determinant approaches zero. The logarithmic penalty function to achieve this, is defined as

$$\mathrm{c}_r(\boldsymbol{\Omega}) := -\tfrac{1}{n\ln n}\ln\det(\tfrac{1}{l}\boldsymbol{\Omega}^\top\boldsymbol{\Omega}). \tag{2.23}$$

The normalization factors with respect to n and l mediate the magnitude of the function value for differently sized $\boldsymbol{\Omega}$.

Furthermore, the coherence of rows of $\boldsymbol{\Omega}$ is controlled by preventing the square of their inner products $(\boldsymbol{\omega}_i^\top\boldsymbol{\omega}_j)^2$, $i \neq j$ from approaching 1 through a second log-barrier function, defined as

$$\mathrm{c}_c(\boldsymbol{\Omega}) := -\tfrac{2}{l(l-1)}\sum_{1\leq i<j\leq l}\ln(1-(\boldsymbol{\omega}_i^\top\boldsymbol{\omega}_j)^2). \tag{2.24}$$

The scaling of the function is again a normalization with respect to the numbers of rows.

Geometrically interpreted, this repulses each row from all others, preventing them from becoming trivially linear dependent. Although a UNTF minimizes c_r and c_c, this formulation is less strict.

Finally, the combination of these penalty functions with the sparsity objective and the oblique manifold constraint, constitutes the GOAL problem

$$\boldsymbol{\Omega}^\star = \underset{\boldsymbol{\Omega}^\top \in \mathrm{OB}(n,l)}{\operatorname{argmin}} \tfrac{1}{2M} \sum_{i=1}^{M} \left(\tfrac{1}{p} \|\boldsymbol{\Omega} \boldsymbol{y}_i\|_p^p \right)^2 + \gamma_r \, \mathrm{c}_r + \gamma_c \, \mathrm{c}_c \,. \tag{2.25}$$

The weight parameters γ_r and γ_c are used to trade the sparsity of the solution with the strictness of the constraints and therefore control mutual coherence and conditioning of $\boldsymbol{\Omega}$. In [74], they are set heuristically. By smoothing the ℓ_p-term at the origin, the non-convex learning objective Eq. (2.25) becomes differentiable and is optimized with an iterative Conjugate Gradient (CG) method [77]. To force iterates $\hat{\boldsymbol{\Omega}}$ to adhere to the oblique manifold, a geometric adaptation of the CG method [10] is used. Instead of updating $\hat{\boldsymbol{\Omega}}$ along the gradient of Eq. (2.25) in the ambient space $\mathbb{R}^{n \times l}$, the geometric gradient descent moves along *geodesics* of the manifold. A more detailed explanation of the geometric adaptation of the CG method to the product manifold of spheres is found in Appendix A.

Experiments on operator recovery conducted in [75] show that GOAL is able to reliably recover analysis operators for approximately co-sparse signals and even outperforms Analysis K-SVD and CAOL.

Sparsifying transform learning

Finally, Ravishankar and Bresler consider a mildly different formulation of the noisy signal analysis model in [145], coined the *transform model*. There, error in modeling a co-sparse approximation of noisy signals is formulated in the representation domain instead of the signal domain. In contrast to the previous considerations, where it was assumed that $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n}$ with co-sparse $\boldsymbol{\Omega}\boldsymbol{x}$, the transform model defines $\boldsymbol{\Omega}\boldsymbol{y} = \boldsymbol{q} + \boldsymbol{\xi}$, where $\boldsymbol{q}$ is co-sparse and $\boldsymbol{\xi}$ denotes the model error in the representation domain. In this way, the representation $\boldsymbol{q}$ is not required to lie exactly in the range space of $\boldsymbol{\Omega}$. In that sense, the transform model can

be considered less restrictive than the noisy signal analysis model. As a consequence, the learning problem for the sparsifying transform, as the authors call the operator $\boldsymbol{\Omega}$, deviates from the previous analysis operator learning problem. One shortcoming of this method is its restriction to only learn square operators $\boldsymbol{\Omega} \in \mathbb{R}^{n \times n}$, which simplifies several steps in its learning procedure. The sparsifying transform learning problem is formulated as

$$\begin{aligned} (\boldsymbol{\Omega}^\star, \boldsymbol{X}^\star) = \underset{\boldsymbol{\Omega}, \boldsymbol{X}}{\operatorname{argmin}} \quad & \|\boldsymbol{\Omega}\boldsymbol{Y} - \boldsymbol{X}\|_F^2 - \gamma_r \log\det \boldsymbol{\Omega} + \gamma_s \|\boldsymbol{\Omega}\|_F^2 \\ \text{subject to} \quad & \|\boldsymbol{X}\|_0 \leq s. \end{aligned} \tag{2.26}$$

Analogously to GOAL, the full-rank constraint is modeled using a similar log-barrier on the operators determinant. Since the operator is restricted to be square, this constraint is imposed directly on the operator. The authors show that for square and full-rank $\boldsymbol{\Omega}$, the function $\|\boldsymbol{\Omega}\boldsymbol{Y} - \boldsymbol{X}\|_F^2 - \gamma_r \log\det \boldsymbol{\Omega}$ is lower-bounded and therefore it is not required to prevent the trivial solution $\boldsymbol{\Omega} = \boldsymbol{0}$ as in the noisy signal analysis operator learning. However, to prevent excessive scaling of individual rows, a penalty on the overall operator Frobenius norm is imposed, which leads exactly to the problem in Eq. (2.26). The weight parameters γ_r and γ_s for the penalty functions are set heuristically.

In parallel to Analysis K-SVD and CAOL, this problem is solved by alternating optimization on $\boldsymbol{X}$ and $\boldsymbol{\Omega}$. In the first step of each iteration k, $\boldsymbol{\Omega}$ is kept fixed and the co-sparse approximation estimate $\hat{\boldsymbol{X}}_k$ is updated by solving $\min_{\boldsymbol{X}} \|\hat{\boldsymbol{\Omega}}_{k-1}\boldsymbol{Y} - \boldsymbol{X}\|_F^2$ subject to $\|\boldsymbol{x}_i\|_0 \leq s$, $\forall i$. The solution is computed either exactly by hard thresholding or by minimizing the soft thresholding surrogate $\min_{\boldsymbol{X}} \|\hat{\boldsymbol{\Omega}}\boldsymbol{Y} - \boldsymbol{X}\|_F^2 + \gamma_t \sum_{i=1}^{M} \|\boldsymbol{x}_i\|_1$ [49]. The update of the operator $\hat{\boldsymbol{\Omega}}_k$ in the second step is achieved while keeping $\hat{\boldsymbol{X}}_k$ fixed and solving the unconstrained, smooth and non-convex function $\min_{\boldsymbol{\Omega} \in \mathbb{R}^{n \times n}} \|\boldsymbol{\Omega}\boldsymbol{Y} - \boldsymbol{X}\|_F^2 - \gamma_r \log\det \boldsymbol{\Omega} + \gamma_s \|\boldsymbol{\Omega}\|_F^2$. It is solved numerically by a CG method with fixed or adaptive step size with backtracking line search. After initialization of $\boldsymbol{\Omega}$ with a random matrix with positive determinant, the two steps are iterated until a standard stopping criterion is met. One advantage of this method is its reduced complexity, which is achieved also due to the restriction to square operators. In a recent extension by Wen et al. [177], the sparsifying transform method was extended to learn over-complete $\boldsymbol{\Omega}$ as a collection of multiple square operators.

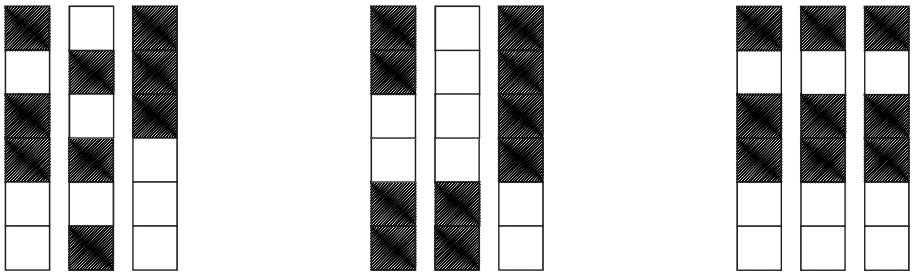
2.2 Patterns in sparse representations

In the discussion of co-sparse representations so far, only overall sparsity as the co-support *cardinality* of the representation was considered. However, it is reasonable to assume that the distribution of zero and non-zero coefficients is not arbitrary across different samples. Rather, since the analysis model is assumed to encode signal subspaces in its representation, it seems imperative that the patterns in the representation coefficients are indeed structured. Further, since the order of rows of $\boldsymbol{\Omega}$ can be permuted in any random way without changing the sparsity assumptions of the model, imposing patterns in the coefficients during signal recovery and analysis operator learning is known as *structured sparsity*.

Let $\boldsymbol{Z} = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_M] \in \mathbb{R}^{l \times M}$ again be a sparse representation matrix for a collection of M signals and let $\mathcal{G}$ be a partitioning of disjoint groups $\mathcal{G} = \{g_1, g_2, \ldots, g_m\}$ of the coefficient indexes in $\boldsymbol{Z}$. Structured sparsity then implies, that coefficients of $\boldsymbol{z}$ that belong to the same group g are simultaneously zero $\boldsymbol{z}^g = \boldsymbol{0}$ or non-zero $\boldsymbol{z}^g \neq \boldsymbol{0}$ within a single $\boldsymbol{z}$ or even across multiple columns in $\boldsymbol{Z}$. Consider the example in Figure 2.1c. There, $g_1 = \{1, 3, 4\}$ and $g_2 = \{2, 5, 6\}$, such that $\boldsymbol{z}_i^{g_1} \neq \boldsymbol{0}$ and $\boldsymbol{z}_i^{g_2} = \boldsymbol{0}$.

Various patterns in the partitioning $\mathcal{G}$ are imaginable and have been proposed in the literature. See Figure 2.1 for an illustration of the most common types. They are mainly motivated in two ways: first, there may exist some dependency within the sparsifying transformation, that causes several elements of the representation to jointly become zero or non-zero. One

Figure 2.1: Illustration of different sparsity patterns. The columns depict samples of sparse representations and their coefficients. The shaded squares indicate non-zero and the white squares zero coefficients.



(a) unstructured sparsity

(b) group sparsity

(c) joint sparsity

such early example arose in wavelet analysis where neighborhood relationships were established among wavelets that share spatial support either in the same scale or across scales [72]. Second, the data samples are obtained in a way that additional dependencies among disparate sections of the signals are imposed. One example is that different data views of the same underlying phenomenon are contained within one sample signal as a concatenation. Consequently, the representation coefficients expose similar patterns associated with the different concatenated sections in the signal domain. Real world instances of such data are found in sound source localization [122], sensor networks [152] or sensor fusion [165], where the same physical phenomenon is measured in different locations, at different times or in different modalities, leading to a distinct pattern.

Certainly, the design of groups in the partitioning $\mathcal{G}$ arise from knowledge about a certain a-priori dependence in the data, whereas there may be additional structure in the representation of a single group that arises from hidden or unknown patterns in the signal itself. It has been shown in several works that if a-priori information suggests that a problem solution can be explained by certain groups of variables, then including a regularization function that automatically selects the relevant groups has proven to yield superior results with respect to predictive performance and interpretability [168, 190, 129, 81, 177].

The methods developed in the context of this thesis rely on patterns in the representation coefficients as well. While other types of patterns are equally important in developing well-suited sparsity methods, the focus here lies on *joint co-sparsity*.

Jointly co-sparse representations

Joint co-sparsity is a specific sparsity pattern [107]. More precisely, two vectors $\boldsymbol{z}_1, \boldsymbol{z}_2 \in \mathbb{R}^l$ are called *jointly co-sparse*, if not only many of their coefficients are zero but at the same time, the indexes of their zero-coefficients coincide. In other words, their co-supports are identical $\operatorname{cosupp}(\boldsymbol{z}_1) = \operatorname{cosupp}(\boldsymbol{z}_2)$. If $\boldsymbol{Z} = [\boldsymbol{z}_1, \boldsymbol{z}_2]$, one requires a sparsity measure that is sensitive to the pattern in $\boldsymbol{Z}$. Practical realizations of such measures typically involve mixed norms. The idea of mixed norms is to first compute a norm over the coefficients of each group defined by the partition $\mathcal{G}$ resulting in a scalar value for each group. For joint sparsity, the partition groups are formed by the rows of $\boldsymbol{Z}$. One such row is denoted by

$\tilde{\boldsymbol{z}}_j$, $j \in \{1, \ldots, l\}$. As a result, one can define the joint sparsity measure as a $(q, 0)$-mixed norm for $\boldsymbol{Z}$ as

$$\psi_{q,0}(\boldsymbol{Z}) := \left\| \left[\|\tilde{\boldsymbol{z}}_j\|_q \right]_{j=1}^{l} \right\|_0. \tag{2.27}$$

The key here is that if the magnitude of all coefficients within a row are zero, the "inner" ℓ_q-norm over them will be zero as well. At the same time, the "outer" ℓ_0-pseudo norm measures sparsity over all row norms. Plainly put, $\psi_{q,0}(\boldsymbol{Z})$ is only small if entire rows of $\boldsymbol{Z}$ are zero and therefore provides an effective measure for the aligned co-supports of columns of $\boldsymbol{Z}$. Typical choices for the "inner" norm are $q = 2$ or $q = \infty$ [66, 107, 168].

Equipped with this joint sparsity measure, one can tackle analysis pursuit and analysis operator learning for jointly co-sparse representations. Interestingly, while joint sparsity for the synthesis model has received considerable attention in the recent literature [168, 51, 42, 167, 166, 118, 82, 165], only few works exist in the co-sparse analysis direction.

The ℓ_0-term spells the same computational trouble to the recovery of jointly co-sparse representations as in the unstructured analysis pursuit setting (see Section 2.1.1) and perhaps unsurprisingly, existing works take similar approaches in their approximation strategies. Replacing the ℓ_0-norm with an ℓ_p-norm with $0 < p \leq 1$ and choosing $q = 2$ is proposed in [84, 132, 90]. Recollect from above that $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is a measurement operator and $\boldsymbol{Y} \in \mathbb{R}^{n \times M}$ the observable noisy measurements of noiseless signals $\boldsymbol{X}$ and $\boldsymbol{Z} = \boldsymbol{\Omega X}$. Then, the jointly co-sparse analysis pursuit problem in its unconstrained form is rendered as

$$\boldsymbol{X}^\star = \underset{\boldsymbol{X}}{\operatorname{argmin}} \, \psi_{2,p}(\boldsymbol{\Omega X}) + \tfrac{\lambda}{2} \|\boldsymbol{Y} - \boldsymbol{AX}\|_2^2, \tag{2.28}$$

with a Lagrangian weight parameter λ. This problem is solved in the convex case $p = 1$ with a level set algorithm [84] or an ADMM variant in [90]. The non-convex setting $(0 < p < 1)$ is considered in [132], where it is solved using adapted split Bregman iterations similar to the method developed for the unstructured analysis pursuit [36].
Also the analysis pursuit methods GAP, AIHT and AHTP (see Section 2.1.1) are available in modified versions for the structured setting with only slight adjustments in the co-support update and signal estimation steps [122, 91].

All of these methods make use of fixed and analytically designed analysis operators. Since it

is known that learning analysis operators from data is beneficial in the unstructured sparsity case (see Subsection 2.1.2), it is imperative to explore methods for learning structured co-sparse representations. For the synthesis model, some methods in this direction have been published recently [165, 85, 175, 106]. However, to the best of the author's knowledge, this remains unexplored for the analysis model and one such method is proposed in Chapter 4.

Chapter 3

Centered co-sparse analysis model

This chapter is based on the peer-reviewed publication:

M. Kiechle et al. "A Bimodal Co-sparse Analysis Model for Image Processing". In: *International Journal of Computer Vision* 114.2 (2015), pp. 233–247

The discussion in the previous chapters revealed that the co-sparsity assumption is a powerful criterion to create models of image data and that they have huge potential in applications. The existing methods mostly consider the data as generic image signals. However, when dealing with data in image processing applications, typically a chain of operations is in place to manipulate the information from its physical representation to digital acquisition and preprocessing. As a result, the sub-space in which the image data of interest resides, is shaped not only by the physical phenomenon whose properties one aims to model but also by the processing chain that is specific to a certain application or to the involved data. Knowledge of the acquisition and preprocessing operations is often available and well-understood. Utilizing a-priori information about properties of the signal space that are caused by these operations, therefore has potential to further improve the modeling of co-sparsity methods and to make them even more useful in practice.

This chapter as well as the subsequent two present approaches where a model of a-priori knowledge of well-known image processing applications is incorporated into an analysis operator learning approach. Therein, the potential to improve modeling accuracy and with

Figure 3.1: Example of different affine variations in illumination on the same local image structure from a gray-scale photo (TUM clock tower[1]). The two columns on the right show excerpts with increasing offset (brightness) and scale (contrast) from top to bottom respectively.

it enhance application performance are explored. As the prevalent type of image data, photometric data and its processing is considered in this chapter. Based on the analysis of widely-used methods that address normalization of image illumination and contrast, an extension to the co-sparse analysis model is proposed, which makes the learning scheme invariant under affine variations in illumination of the image data. In particular, the GOAL method from [74] is modified to incorporate this extension.

3.1 Brightness and contrast variations in image data

To an observer, brightness and contrast of a photo are obvious features and a well-versed photographer knows how to manipulate these *illumination* properties. Although they have an impact on the overall impression of an image, they are mostly ignored for the analysis of local image structures in the literature and typically normalized explicitly (see e.g. [74]). Figure 3.1 illustrates a photo under different brightness and contrast settings.

[1] Photograph by Andreas Heddergott, TUM Corporate Communications Center. Permission to use granted on 20.12.2018.

To formulate a co-sparse analysis model that is invariant to brightness and contrast, their description is formalized first. To that end, the well-established bias-and-gain illumination model for images [162] is considered. It expresses variance in brightness and contrast by an affine transformation of the pixel intensities. Let $\boldsymbol{u} \in \mathbb{R}^n$ denote a vectorized gray-scale image of size $h \times w$ by arranging its pixels in lexicographical order, hence $n = hw$. Let α, β further be two scalar quantities, then any image signal of pixel intensities $\boldsymbol{u}$ can be written as

$$\boldsymbol{u} = \alpha \bar{\boldsymbol{u}} + \beta. \tag{3.1}$$

The vector $\bar{\boldsymbol{u}}$ denotes an image whose pixel intensity mean equals zero, i.e. $\sum_{i=1}^{n} \bar{u}_i = 0$, and will be referred to as the *centered* or *brightness normalized* image. By varying the parameters α and β, one can now adjust brightness and contrast of the image. Conversely, if one is interested in the intrinsic image structure independently of illumination variations, it is desirable to remove the effect of α and β, particularly so, if the structure of a collection of images is supposed to be modeled independently of brightness or contrast. Denoting the identity matrix of size $n \times n$ by $\boldsymbol{I}_n$ and the same-sized matrix whose coefficients all equal one by $\boldsymbol{J}_n$, the removal of bias and gain from the image vector $\boldsymbol{u}$ can be formulated as two operations. First, centering the signal by removing its coefficient mean

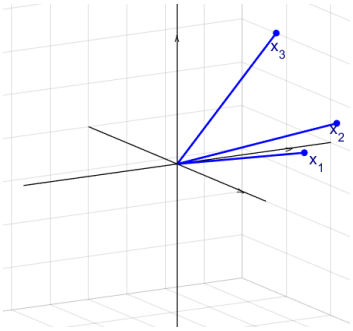
$$\boldsymbol{u}_\alpha = (\boldsymbol{I}_n - \tfrac{1}{n}\boldsymbol{J}_n)\,\boldsymbol{u}, \tag{3.2}$$

followed by a normalization of the scale

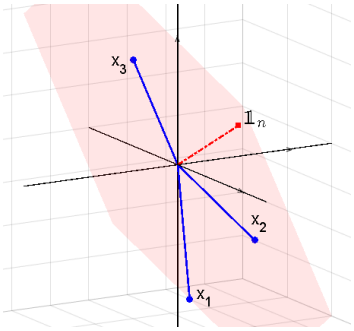
$$\bar{\boldsymbol{u}} = \begin{cases} \frac{\boldsymbol{u}_\alpha}{\|\boldsymbol{u}_\alpha\|_2}, & \|\boldsymbol{u}_\alpha\|_2 \neq 0 \\ \boldsymbol{0}, & \|\boldsymbol{u}_\alpha\|_2 = 0 \end{cases} \tag{3.3}$$

This pre-processing is widely used in various image processing applications [144]. One can observe that this transformation of the pixel intensities reduces the dimensionality of the data. The reduction is caused once by projecting the vector onto the hyperplane whose normal is constant and a second time when projecting the resulting vector onto the sphere by scaling with its norm. Figure 3.2 depicts this for three-dimensional data points. Low-level image models capture the structure of image patches which collectively comprise an image. Since the normalizations are performed on the image patches, many of such atomic structures are highly redundant in a single image and even more so on a larger set of images.

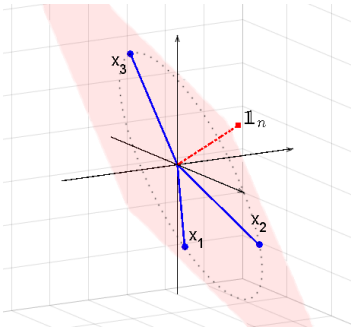
Figure 3.2: Effect of illumination normalization from Eq. (3.2) and (3.3) on the geometry of vectorized image patches with positive pixel intensities. The original patch vectors (a) are projected into a lower-dimensional subspace, illustrated by (b) and (c) respectively. The hyperplane of centered patches is shaded in red with its normal vector ($\mathbb{1}_n$); the subspace of centered and unit-norm patches is indicated by the dotted line.



(a) vectorized image patches of three pixels



(b) patch vectors with centered pixel values



(c) centered and unit-norm patch vectors

As a result, the normalization leads to higher redundancy in the data. One explanation why all of the related methods discussed in Chapter 2 perform such normalizations on the training data is that the models learned in this way generalize better due to the illumination invariant training.

Since one is interested in the inherent structure of the image independent of contrast and brightness, using zero-mean patches with unit scale for learning image structure is the most natural choice. Another consequence of this pre-processing is that the dimensionality-reduced data introduces new trivial solutions in the analysis operator learning problem. Although often performed in practical experiments, the effect of data normalization on the training data has not been addressed in previous learning frameworks for the co-sparse analysis model. The following model for learning co-sparse representations aims to rectify these issues.

3.2 Co-sparse analysis model with centered rows

Recollect from the first chapters, that the co-sparse analysis model assumes that for a given class of signals $\mathrm{X} \subset \mathbb{R}^n$, there exists an analysis operator $\boldsymbol{\Omega} \in \mathbb{R}^{l \times n}$ such that the analyzed

vector $\boldsymbol{\Omega x}$ is sparse for all $\boldsymbol{x} \in \mathrm{X}$. Geometrically, X is contained in a union of subspaces and $\boldsymbol{x} \in \mathrm{X}$ lies in the intersection of all hyperplanes whose normal vectors are given by the rows of $\boldsymbol{\Omega}$ that are indexed by the zero entries of $\boldsymbol{\Omega x}$, called the co-support.

If the affine transformation of the bias-and-gain model from Eq. (3.1) is now considered for an image signal, scale and offset influence its analyzed vector

$$\boldsymbol{z} := \boldsymbol{\Omega x} = \boldsymbol{\Omega}(\alpha \bar{\boldsymbol{x}} + \beta \mathbb{1}_n) = \alpha \boldsymbol{\Omega} \bar{\boldsymbol{x}} + \beta \boldsymbol{\Omega} \mathbb{1}_n. \tag{3.4}$$

Here, $\mathbb{1}_n$ denotes a vector of size n whose coefficients all equal one. It is apparent from Eq. (3.4), that if the signal offset β is not zero, a coefficient of the analyzed vector $(\boldsymbol{\Omega x})_j$ is determined by the inner product of the corresponding analysis operator row and the centered signal $\boldsymbol{\omega}_j^\top \bar{\boldsymbol{x}}$ *and* the sum of the operator row $\boldsymbol{\omega}_j^\top \mathbb{1}_n = \sum_i \omega_{j,i}$, scaled by β. The gain α of $\boldsymbol{x}$ is dealt with naturally, since the model-induced sparsity is oblivious to the scale of the analyzed vector coefficients, i.e. $\|\boldsymbol{z}\|_0 = \|\alpha \boldsymbol{z}\|_0$. The offset β, on the other hand, tampers with the sparsity and clearly, to encode the structure of $\boldsymbol{x}$ invariantly of its bias, each row of the analysis operator individually needs to sum up to zero.

$$\sum_{i=1}^{n} \omega_{j,i} = 0 \quad 1 \leq j \leq l. \tag{3.5}$$

Certainly, if an analytic operator is employed, this requirement is of little use, since its coefficients are fixed and removing the mean of each row may even decrease its ability to produce sparse representations. If the analysis operator is learned, however, this constraint should be included in the corresponding optimization task.

Geometric analysis operator learning with centered rows

Ad-hoc models are inferior to models that are adapted to the specific class X of interest, cf. [150, 185, 74, 145] and *analysis operator learning* aims to find the most suitable analysis operator for a given class X of signals. Here and in the remainder of this thesis, the focus lies on the flavor of geometric analysis operator learning as proposed in [74]. Recollecting from Chapter 2, the learning is achieved by optimizing overall sparsity over a representative

set of training samples $\{\boldsymbol{x}_i\}_{i=1}^M \subset \mathrm{X}$, solving

$$\boldsymbol{\Omega}^\star \in \arg\min_{\boldsymbol{\Omega}} \sum_i^M \|\boldsymbol{\Omega}\boldsymbol{x}_i\|_0. \tag{3.6}$$

This problem requires a useful set of constraints in order to prevent the following undesirable solutions:

1. $\boldsymbol{\Omega} = \mathbf{0}$ is a minimizer of Eq. (3.6) but does not encode any structure of the signal.
2. $\boldsymbol{\Omega}$ can become rank deficient and therefore discard information contained in analyzed signals.
3. even when $\boldsymbol{\Omega}$ has full rank, it may contain duplicate or trivially linear dependent rows, i.e. $\boldsymbol{\omega}_i = \pm\boldsymbol{\omega}_j, i \neq j$, that encode the same information about an analyzed signal.

As detailed in Section 2.1.2, GOAL prevents these trivial solutions by a norm constraint on the rows $\boldsymbol{\omega}$ as well as log-barrier penalties on the determinant of the Gramian $\boldsymbol{\Omega}^\top\boldsymbol{\Omega}$ and the pairwise inner products of rows. To learn analysis operators from data whose rows are *centered*, the GOAL constraint set needs to be revisited.

Row norm constraint

In a first step, the constraint that rows of $\boldsymbol{\Omega}$ need to have unit Euclidean norm is considered, i.e. the restriction of the transpose of possible solutions, the so-called *oblique manifold*

$$\boldsymbol{\Omega}^\top \in \mathrm{OB}(n, l) := \mathbb{S}_{n-1}^{\times l}, \tag{3.7}$$

where $\mathbb{S}_{n-1}$ denotes the unit sphere in $\mathbb{R}^n$.

In the previous section it was shown that the subtraction of the signal mean in Eq. (3.2) is equal to projecting the signal onto the orthogonal complement of $\mathbb{1}_n$ which is denoted by $\mathbb{1}_n^\perp$. This operation, however, introduces another trivial solution $\boldsymbol{\omega} = \frac{1}{\sqrt{n}}\mathbb{1}_n$ for the rows of the operator $\boldsymbol{\Omega}$ and which does not encode any useful information. Therefore, the previous

approach is extended by further restricting the rows of possible solutions to the orthogonal complement of $\mathbb{1}_n$. A similar idea was remarked in [185]. As a result, the transposed of admissible solutions $\boldsymbol{\Omega}$ is contained in the intersection of the two sets formed by the oblique manifold and the orthogonal space of $\mathbb{1}_n$, i.e.

$$\mathcal{R} := \left(\mathbb{S}_{n-1} \cap \mathbb{1}_n^{\perp}\right)^{\times l}. \tag{3.8}$$

It will be shown in Section 3.3 that $\mathcal{R}$ is in fact a smooth Riemannian manifold which allows us to elegantly prevent any trivial solution by using an appropriate geometric gradient algorithm.

Full-rank and coherence constraints

In addition to avoiding trivial solutions, it has been well investigated, e.g. in [73, 185], that coherence and rank are important properties to control for finding analysis operators that well represent a signal class. In the setting here, where permissible analysis operators are learned on a subspace, the necessary constraints for restricting rank and coherence properties need to be revisited.

The approach by Hawe et al. is adopted, where the rank of $\boldsymbol{\Omega}$ is regularized with the penalty function in Eq. (2.23)

$$-\tfrac{1}{n \ln n} \ln \det(\tfrac{1}{l} \boldsymbol{\Omega}^\top \boldsymbol{\Omega}).$$

Since $\boldsymbol{\Omega}$ is now constrained to a lower-dimensional subspace, this penalty can not be satisfied anymore. To that end, the argument of the log-barrier is extended by an orthonormal projection of $\boldsymbol{\Omega}$ as follows

$$h(\boldsymbol{\Omega}) := -\tfrac{1}{(n-1)\log(n-1)} \log \det(\tfrac{1}{l} \boldsymbol{W}^\top \boldsymbol{\Omega}^\top \boldsymbol{\Omega} \boldsymbol{W}), \tag{3.9}$$

in which the columns of $\boldsymbol{W} \in \mathbb{R}^{n \times (n-1)}$ form an arbitrary orthonormal basis of $\mathbb{1}_n^{\perp}$. This adjustment to the original penalty accounts for the rank deficiency of $\boldsymbol{\Omega}$ imposed by the new manifold setting from Eq. (3.8).

The constraint on row coherence from Eq. (2.24) which is formed by a log-barrier function

on the scalar product of all rows of the operator, and which enforces distinct rows, i.e.

$$r(\boldsymbol{\Omega}) := -\sum_{1\leq m<n\leq l} \log(1-(\boldsymbol{\omega}_m^\top \boldsymbol{\omega}_n)^2), \tag{3.10}$$

does not require any adjustment, since its semantics are invariant to the changed space. However, since the same number of rows is now contained in a smaller space, the value of this penalty function is expected to be larger in magnitude.

Relaxation of strict co-sparsity

An important aspect of sparsity methods is the computational tractability of the sparsity measure and all of the prevailing strategies from the literature are discussed in Chapter 2. Clearly, the model in Eq. (3.6) is idealized, since in practice the entries of the analyzed vectors are not exactly equal to zero but small in magnitude. Here, the strategy of a non-convex relaxation of the ℓ_0-pseudo-norm is chosen and the smooth log-square function

$$g\colon \mathbb{R}^l \to \mathbb{R}_+ \colon \boldsymbol{z} \mapsto \frac{1}{\log(1+\nu)} \sum_{j=1}^{l} \log(1+\nu z_j^2), \tag{3.11}$$

is used in its place. Here, z_j denotes the j-th entry of $\boldsymbol{z}$. Similar to the log-sum sparsity measure proposed in [31], the log-square function in Eq. (3.11) is – up to a constant factor – a good approximation of ℓ_0-sparsity for large values of ν. Indeed, using l'Hôpital's rule, it is verified that

$$\lim_{\nu\to\infty} \frac{1}{\log(1+\nu)} \log(1+\nu x^2) = \begin{cases} 1 & x \neq 0 \\ 0 & x = 0 \end{cases} \tag{3.12}$$

Using the log-square sparsity measure leads to the relaxed co-sparse analysis operator learning formulation

$$\boldsymbol{\Omega}^\star \in \arg\min_{\boldsymbol{\Omega}} \sum_i g(\boldsymbol{\Omega}\boldsymbol{x}_i). \tag{3.13}$$

Despite being non-convex, the log-square term has favorable properties such as smoothness and close resemblance to ℓ_0-sparsity, as shown in Figure 3.4.

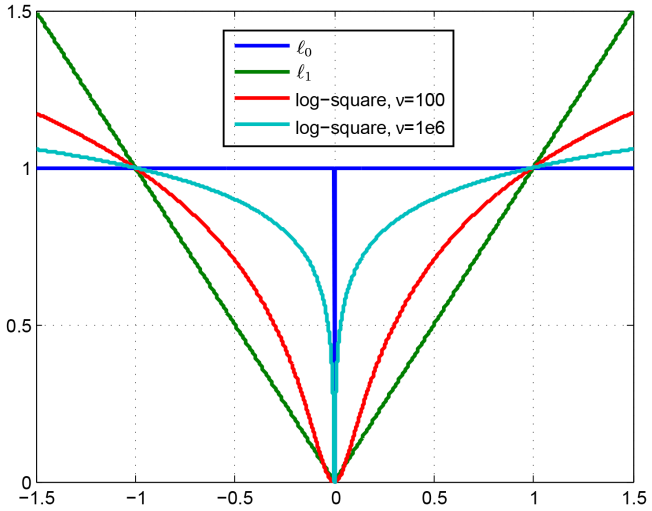


Figure 3.4: Plot of the ℓ_0, ℓ_1 and log-square sparsity functions.

For learning the analysis operator from data, the training samples are expected to be equally important. If Eq. (3.13) is minimized, however, it is possible and even likely that the found solution will encode the training set unevenly, yielding some analyzed vectors with very large co-sparsity while others are only mildly sparse. To avoid an artificial bias during training, one may instead minimize variance and mean of the sparsity over the entire training set [74].

Let $\bar{g}^2$ denote the squared mean of the sparsity measure

$$\bar{g}^2 = \left(\frac{1}{M} \sum_i g(\boldsymbol{\Omega} \boldsymbol{x}_i) \right)^2 \tag{3.14}$$

and σ_g^2 its empirical variance

$$\sigma_g^2 = \frac{1}{M} \sum_i \left(g(\boldsymbol{\Omega} \boldsymbol{x}_i) - \bar{g} \right)^2 , \tag{3.15}$$

then the sum of both writes as [75]

$$\begin{aligned}
\bar{g}^2 + \sigma_g^2 &= \bar{g}^2 + \frac{1}{M}\sum_i \left(g(\boldsymbol{\Omega}\boldsymbol{x}_i) - \bar{g}\right)^2 \\
&= \bar{g}^2 + \frac{1}{M}\sum_i \left(g(\boldsymbol{\Omega}\boldsymbol{x}_i)^2 - 2g(\boldsymbol{\Omega}\boldsymbol{x}_i)\bar{g} + \bar{g}^2\right) \\
&= \bar{g}^2 + \frac{1}{M}\sum_i g(\boldsymbol{\Omega}\boldsymbol{x}_i)^2 - \frac{2}{M}\sum_i g(\boldsymbol{\Omega}\boldsymbol{x}_i)\bar{g} + \frac{1}{M}\sum_i \bar{g}^2 \\
&= 2\bar{g}^2 + \frac{1}{M}\sum_i g(\boldsymbol{\Omega}\boldsymbol{x}_i)^2 - \frac{2}{M}\sum_i g(\boldsymbol{\Omega}\boldsymbol{x}_i)\bar{g} \\
&= 2\bar{g}^2 + \frac{1}{M}\sum_i g(\boldsymbol{\Omega}\boldsymbol{x}_i)^2 - 2\bar{g}^2 \\
&= \frac{1}{M}\sum_i g(\boldsymbol{\Omega}\boldsymbol{x}_i)^2
\end{aligned} \tag{3.16}$$

and is used instead of g to model an analysis operator that yields co-sparse representations that are balanced across all M samples.

Learning objective for analysis operators with centered rows

The final learning function for analysis operators with centered rows is constructed by combining the smoothed sparsity objective from Eq. (3.13) with the constraint set, which is formed by the manifold in Eq. (3.8) and the penalty functions from Eq. (3.9) and Eq. (3.10).

Let $\kappa, \mu \in \mathbb{R}_+$ denote positive weights that determine the strictness of rank and coherence constraints over the sparsity of the analyzed training samples. Then, the optimization problem for learning analysis operators with centered rows from data samples reads as

$$\boldsymbol{\Omega}^\star \in \operatorname*{arg\,min}_{\boldsymbol{\Omega}^\top \in \mathcal{R}} L(\boldsymbol{\Omega}), \qquad L(\boldsymbol{\Omega}) := \frac{1}{M}\sum_{i=1}^{M} g(\boldsymbol{\Omega}\boldsymbol{x}_i)^2 + \kappa h(\boldsymbol{\Omega}) + \mu r(\boldsymbol{\Omega}). \tag{3.17}$$

The problem is non-convex due to the involved sparsity measure and the non-convex admissible set defined by the manifold $\mathcal{R}$. However, the objective function is smooth and

continuous and allows us to design an iterative gradient-based solver. Furthermore, the geometric structure of the problem imposed by the manifold constraint can be exploited by the solver to compute solutions to Eq. (3.17) efficiently.

3.3 Learning model parameters from data

In order to learn useful parameters $\boldsymbol{\Omega}$ for the problem in Eq. (3.17) from training data, an optimization method suitable for the smooth but non-convex learning function is required, while being able to cope with the geometric constraint that is imposed by the manifold geometry. The geometric conjugate gradient method, as explained in Appendix A, is an excellent candidate and has also been used in similar optimization problems, e.g. [74, 76, 178, 127]. However, the conjugate gradient solver used in [74] can not readily be applied to the manifold structure in Eq. (3.8) and needs to be appropriately adjusted.

Adapted geometric CG on the sphere

For learning the model parameters of Eq. (3.17), a geometric conjugate gradient method adapted to the manifold structure $\mathcal{R}$ is proposed and will be further referred to as GeOmetric Analysis operator Learning with Centered rows (C-GOAL). First, one has to ensure that $\mathcal{R}$ as given in Eq. (3.8) is indeed a manifold.

Lemma. The set $\mathcal{R} = \left(\mathbb{S}_{n-1} \cap \mathbb{1}_n^\perp\right)^{\times l}$ is a Riemannian sub-manifold of $\mathbb{R}^{n\times l}$ and the tangent space at $\boldsymbol{O} = [\boldsymbol{o}_1, \ldots, \boldsymbol{o}_l] \in \mathcal{R}$ is given by

$$T_{\boldsymbol{O}}\mathcal{R} = T_{\boldsymbol{o}_1}(\mathbb{S}_{n-1} \cap \mathbb{1}_n^\perp) \times \cdots \times T_{\boldsymbol{o}_l}(\mathbb{S}_{n-1} \cap \mathbb{1}_n^\perp), \tag{3.18}$$

with

$$T_{\boldsymbol{o}}(\mathbb{S}_{n-1} \cap \mathbb{1}_n^\perp) = \left\{\boldsymbol{h} \in \mathbb{R}^n \mid \boldsymbol{h}^\top [\boldsymbol{o}, \mathbb{1}_n] = \boldsymbol{0}\right\}. \tag{3.19}$$

Proof. By using the product manifold structure, it is sufficient to show that $\mathbb{S}_{n-1} \cap \mathbb{1}_n^\perp$ is

a sub-manifold of $\mathbb{R}^n$ with its tangent space as given in Eq. (3.19). Consider the function

$$F\colon \mathbb{R}^n \to \mathbb{R}^2, \quad \boldsymbol{o} \mapsto \begin{bmatrix} \|\boldsymbol{o}\|^2 - 1 \\ \boldsymbol{o}^\top \mathbb{1}_n \end{bmatrix}. \tag{3.20}$$

Then $\mathbb{S}_{n-1} \cap \mathbb{1}_n^\perp = F^{-1}(0)$ and the derivative of F is given by

$$DF(\boldsymbol{o})\boldsymbol{h} = \begin{bmatrix} 2\boldsymbol{o}^\top \\ \mathbb{1}_n^\top \end{bmatrix} \boldsymbol{h}, \tag{3.21}$$

which is surjective for all $\boldsymbol{o} \in \mathbb{S}_{n-1} \cap \mathbb{1}_n^\perp$. The regular value theorem now implies that $\mathbb{S}_{n-1} \cap \mathbb{1}_n^\perp$ is a sub-manifold of $\mathbb{R}^n$ and that $T_{\boldsymbol{o}}(\mathbb{S}_{n-1} \cap \mathbb{1}_n^\perp)$ is given by the null space of $DF(\boldsymbol{o})$, yielding Eq. (3.19). □

With respect to the standard inner product, the orthogonal projection onto $T_{\boldsymbol{o}}(\mathbb{S}_{n-1} \cap \mathbb{1}_n^\perp)$ is given by the projection matrix

$$\boldsymbol{P}_{\boldsymbol{o}} = \left(\boldsymbol{I}_n - \boldsymbol{Q}_{\boldsymbol{o}}\boldsymbol{Q}_{\boldsymbol{o}}^\top\right), \tag{3.22}$$

where

$$\boldsymbol{Q}_{\boldsymbol{o}} = \left[\boldsymbol{o}, \tfrac{1}{\sqrt{n}}\mathbb{1}_n\right] \tag{3.23}$$

has orthonormal columns and $\boldsymbol{I}_n$ is the identity matrix. Using the product manifold structure, one finds the orthogonal projection of an element $\boldsymbol{Y} \in \mathbb{R}^{n\times l}$ onto $T_{\boldsymbol{O}}\mathcal{R}$ as

$$\Pi_{\boldsymbol{O}}\left[\boldsymbol{y}_1, \ldots, \boldsymbol{y}_l\right] = \left[\boldsymbol{P}_{\boldsymbol{o}_1}\boldsymbol{y}_1, \ldots, \boldsymbol{P}_{\boldsymbol{o}_l}\boldsymbol{y}_l\right]. \tag{3.24}$$

In order to compute the Riemannian gradient of the learning function in Eq. (3.17), its Euclidean gradient with respect to the standard inner product needs to be available. The learning function L is smooth and the gradient is obtained from the derivatives with respect to analysis operator, given by

$$\nabla L(\boldsymbol{\Omega}) = \nabla f(\boldsymbol{\Omega}) + \kappa\nabla h(\boldsymbol{\Omega}) + \mu\nabla r(\boldsymbol{\Omega}). \tag{3.25}$$

The first summand of the learning function is the sparsity term. For convenience, let $\boldsymbol{Z} = \boldsymbol{\Omega}\boldsymbol{X}$ again be the co-sparse codes of the M training samples $\boldsymbol{X} = [\boldsymbol{x}_i]_{i=1}^M$ and let z_{ji} be its coefficient in the j-th row of the i-th column, then the sparsity term is defined as

$$\begin{aligned} f(\boldsymbol{\Omega}) &:= \frac{1}{M}\sum_{i=1}^{M} g(\boldsymbol{\Omega}\boldsymbol{x}_i)^2 \\ &= \frac{1}{M}\sum_{i=1}^{M}\left[\frac{1}{\log(1+\nu)}\sum_{j=1}^{l}\log(1+\nu z_{ji}^2)\right]^2 \\ &= \frac{1}{M}\frac{1}{\log(1+\nu)^2}\sum_{i=1}^{M}\left[\sum_{j=1}^{l}\log(1+\nu z_{ji}^2)\right]^2. \end{aligned} \tag{3.26}$$

The gradient of the sparsity term $f(\boldsymbol{\Omega})$ is written as

$$\nabla f(\boldsymbol{\Omega}) = \frac{2}{M}\frac{1}{\log(1+\nu)^2}\left[\sum_{i=1,j=1}^{M,l}\frac{2\nu z_{ji}\left(\sum_{j=1}^{l}\log(1+\nu z_{ji}^2)\right)^2}{1+\nu z_{ji}^2}\boldsymbol{J}_{ji}\right]\boldsymbol{X}^\top. \tag{3.27}$$

The matrix $\boldsymbol{J}_{ji} \in \mathbb{R}^{l\times M}$ is used for notation clarity. All of its elements are zero except for the j-th element in the i-th row, which equals one.

To derive the gradient of the rank penalty function $h(\boldsymbol{\Omega})$ in Eq. (3.9), it is helpful [75] to rewrite its formulation as

$$\begin{aligned} h(\boldsymbol{\Omega}) &= -\tfrac{1}{(n-1)\log(n-1)}\log\det(\tfrac{1}{l}\boldsymbol{W}^\top\boldsymbol{\Omega}^\top\boldsymbol{\Omega}\boldsymbol{W}) \\ &= -\tfrac{1}{(n-1)\log(n-1)}\log(\prod_{j=1}\tfrac{1}{l}\lambda_j) \\ &= -\tfrac{1}{(n-1)\log(n-1)}\log(\tfrac{1}{l^{(n-1)}}\prod_{j=1}\lambda_j) \\ &= \tfrac{\log(l)}{\log(n-1)} - \tfrac{1}{(n-1)\log(n-1)}\log\det(\boldsymbol{W}^\top\boldsymbol{\Omega}^\top\boldsymbol{\Omega}\boldsymbol{W}), \end{aligned} \tag{3.28}$$

with the $n-1$ eigenvalues of $\boldsymbol{W}^\top\boldsymbol{\Omega}^\top\boldsymbol{\Omega}\boldsymbol{W}$ denoted as λ_j.

Deriving the gradient of $h(\boldsymbol{\Omega})$ now results in

$$\nabla h(\boldsymbol{\Omega}) = -\frac{2}{(n-1)\log(n-1)} \boldsymbol{\Omega}\boldsymbol{W} \left(\boldsymbol{W}^\top \boldsymbol{\Omega}^\top \boldsymbol{\Omega}\boldsymbol{W}\right)^{-1} \boldsymbol{W}^\top. \tag{3.29}$$

Finally, the gradient of the coherence penalty function $r(\boldsymbol{\Omega})$ in Eq. (3.10) is needed. Since the pairwise inner products of analysis operator rows are not affected by the changed manifold structure compared to the oblique manifold, the gradient defined in [74] can be reused directly, i.e.

$$\nabla r(\boldsymbol{\Omega}) = \frac{2}{l(l-1)} \left[\sum_{1 \le i < j \le l} \frac{2\boldsymbol{\omega}_m^\top \boldsymbol{\omega}_n}{1 - (\boldsymbol{\omega}_m^\top \boldsymbol{\omega}_n)^2} (\boldsymbol{J}_{mn} + \boldsymbol{J}_{nm}) \right] \boldsymbol{\Omega}. \tag{3.30}$$

The matrices $\boldsymbol{J}_{mn}, \boldsymbol{J}_{nm} \in \mathbb{R}^{l \times l}$ are used for notation ease as before in Eq. (3.27).

Having derived the Euclidean gradient of the smooth learning function, the Riemannian gradient can finally be obtained using Eq. (3.24) on the transposed Euclidean gradient (since $\boldsymbol{\Omega}^\top \in \mathcal{R}$), and is thus

$$\boldsymbol{G}(\boldsymbol{\Omega}^\top) = \Pi_{\boldsymbol{\Omega}} \nabla L(\boldsymbol{\Omega})^\top. \tag{3.31}$$

The last two missing ingredients to iteratively update estimations to minimize the learning function are the geodesics along which an iterate is updated on $\mathcal{R}$ in Eq. (A.5), and the parallel transport for combining elements of different tangent spaces in Eq. (A.6). Since $\mathcal{R}$ is a sub-manifold of the oblique manifold $\mathrm{OB}(n, l)$ and is thereby also a sub-manifold of the product of l unit spheres $\mathbb{S}^{n-1}$ [74], the equations for the geodesics and parallel transport for $\mathcal{R}$ are the same as for $\mathbb{S}^{n-1}$. Their definitions were derived in [10] as follows. The geodesic through a point $\boldsymbol{o}$ on a sphere $\mathbb{S}^{n-1}$ that is parallel to the tangent vector $\boldsymbol{h} \in T_{\boldsymbol{o}}\mathbb{S}^{n-1}$ in $\boldsymbol{o}$ is a great circle given by

$$\Gamma(\boldsymbol{o}, \boldsymbol{h}, t) = \begin{cases} \boldsymbol{o} & \|\boldsymbol{o}\|_2 = 0 \\ \boldsymbol{o}\cos(\|\boldsymbol{o}\|_2 t) + \frac{1}{\|\boldsymbol{h}\|_2}\boldsymbol{h}\sin(\|\boldsymbol{h}\|_2 t) & \textit{otherwise} \end{cases} \tag{3.32}$$

Based on this definition, the vector transport Ψ which translates an element of the tangent space $\boldsymbol{\xi} \in T_{\boldsymbol{o}}\mathcal{R}$ along the geodesic emanating from $\boldsymbol{o} \in \mathcal{R}$ in the direction of $\boldsymbol{h} \in T_{\boldsymbol{o}}\mathcal{R}$, reads

as

$$\Psi(\boldsymbol{\xi}, \boldsymbol{o}, \boldsymbol{h}, t) := \boldsymbol{\xi} - \frac{1}{\|\boldsymbol{h}\|_2^2} \left[\boldsymbol{\xi}^\top \boldsymbol{h} \left(\boldsymbol{h} + \|\boldsymbol{h}\|_2 \sin(\|\boldsymbol{h}\|_2 t) \boldsymbol{o} \right) - \boldsymbol{\xi}^\top \boldsymbol{o} \|\boldsymbol{h}\|_2 \cos(\|\boldsymbol{h}\|_2 t) \boldsymbol{h} \right]. \quad (3.33)$$

Following the general conjugate gradient scheme outlined in Appendix A, it is now straightforward to implement the learning algorithm.

3.4 Image reconstruction from partial data

A direct comparison of co-sparse analysis operator learning methods is difficult for the proposed approach. Since the dimensionality of the considered data differs depending on the applied pre-processing, comparing the values of the objective function during training or the co-sparsity of the data can not tell, which model is more useful in real applications. A common practice is to use a proxy experiment instead where the learned models are employed to reconstruct images and are compared based on the quality of the reconstruction result. In this section, one such proxy experiment, image reconstruction from partial data is explained and conducted with real image data. To that end, single-channel photometric images are reconstructed from a small number of their pixels by removing all other pixels. The ensued highly ill-posed inverse problem to recover the full image requires a regularizer which is supplied by the learned analysis operators.

Applying the patch-based operators to images

The described learning processes of analysis operators are all fueled by image patches of small extent. Yet, when used as an image regularizer, the operator needs to be applied to images of much larger dimensions. Therefore, a global formulation of the local operators to process images is required first. Consider an image $\boldsymbol{U} \in \mathbb{R}^{h \times w}$ of height h and width w, as well as its vectorized form $\boldsymbol{u} \in \mathbb{R}^{hw}$. According to [73], a *global* analysis operator $\boldsymbol{\Omega}^F \in \mathbb{R}^{K \times N}$ can be constructed for application to an image of size $N = hw$ from a patch-based operator $\boldsymbol{\Omega} \in \mathbb{R}^{l \times n}$ as follows. Denote the operator that extracts a $(\sqrt{n} \times \sqrt{n})$-dimensional patch located at position (r, c) in the large image as $\mathcal{P}_{rc} \in \mathbb{R}^{n \times N}$. The global formulation of the

analysis operator is then given as

$$\boldsymbol{\Omega}^F := \begin{bmatrix} \boldsymbol{\Omega}\mathcal{P}_{11} \\ \boldsymbol{\Omega}\mathcal{P}_{21} \\ \vdots \\ \boldsymbol{\Omega}\mathcal{P}_{hw} \end{bmatrix} \in \mathbb{R}^{K \times N}, \tag{3.34}$$

with $K = lN$, i.e. all patch positions are considered. The reflective boundary condition is used to deal with areas along image borders.

Formulation of the reconstruction problem

Consider again image $\boldsymbol{U} \in \mathbb{R}^{h \times w}$ as well as its vectorized form $\boldsymbol{u} \in \mathbb{R}^{hw}$, of which only a small subset of pixel measurements are available. The masking operator $\boldsymbol{A}$ models this as an extraction of m pixel values from the original image vector to obtain the measurements $\boldsymbol{y} \in \mathbb{R}^m$

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{u}. \tag{3.35}$$

Here, the number of measurements is significantly smaller than in the original image, i.e. $m << hw$, which makes the problem of recovering the original image highly ill-posed. To regularize the reconstruction problem, the learned analysis operators are incorporated into the formulation of the recovery task, requiring that a solution of the reconstructed image $\boldsymbol{x}^\star$ be consistent with the few measurements $\boldsymbol{y}$ but at the same time be co-sparse under the learned analysis operator. Accordingly, the optimization task can be written as

$$\boldsymbol{x}^\star \in \underset{\boldsymbol{x} \in \mathbb{R}^N}{\arg\min} \, \tfrac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda g(\boldsymbol{\Omega}^F \boldsymbol{x}). \tag{3.36}$$

Here again, the relaxed smooth sparsity measure of the log-square function, defined in Eq. (3.11), is used for g. The parameter λ balances the co-sparsity of the solution with its fidelity to the measurements. This recovery function is smooth but non-convex and is solved with a standard conjugate gradient solver that operates in the Euclidean space $\mathbb{R}^N$.

3.5 Experiments

To validate the assumption of the proposed model empirically, a set of analysis operators is trained on training images and subsequently used as regularizers in an image reconstruction task, cast as an inverse problem. This experiment is set to compare how pre-processing of the photometric training data influences the training process of the original GOAL and the proposed model with centered rows as well as tests the obtained operators' performance in a practical reconstruction task.

Analysis operator learning from training images

First, a set of operators is learned from data by minimizing Eq. (3.17) using the algorithm described in Section 3.3. In addition, the method from [74] is used to solve the related GOAL learning problem. Following the methodology in previous works, a set of $M = 35000$ image patches of square size is collected from the set of training images, depicted in Figure 3.5. The patches of width $\sqrt{n} = 8$ are extracted uniformly from pixel locations in the training images, their pixels are vectorized in lexicographical order and scaled to unit length using Eq. (3.3) to comprise the training data $\tilde{\boldsymbol{X}} \in \mathbb{R}^{n \times M}$. In addition, the normalized data $\boldsymbol{X}$ is constructed by applying the centering step from Eq. (3.2). Using GOAL from [74] as well as the proposed method, the parameters of four analysis operators are trained from this data. The first two operators $\hat{\boldsymbol{\Omega}}_{GOAL}$ and $\boldsymbol{\Omega}_{GOAL}$ are trained using the GOAL algorithm, while the other two $\boldsymbol{\Omega}$ and $\boldsymbol{\Omega}_{rr}$ (rr short for 'reduced rows') are trained using the proposed algorithm. The learning of $\hat{\boldsymbol{\Omega}}_{GOAL}$ is based on the raw training data $\tilde{\boldsymbol{X}}$, while the other three operators are trained from the normalized data $\boldsymbol{X}$. The first three operators are trained with double over-completeness $l = 2n$, implying $\boldsymbol{\Omega} \in \mathbb{R}^{128 \times 64}$ due to the chosen patch size. GOAL requires one row to represent signal mean. To make sure that any improvement of the proposed model is

Figure 3.5: Training images used for learning different analysis operators.

not only due to the fact that it does not need to spend one row for signal mean, the additional $\boldsymbol{\Omega}_{rr}$ is trained with a correspondingly reduced row set, i.e. $\boldsymbol{\Omega}_{rr} \in \mathbb{R}^{127\times 64}$. The operators' coefficients are initialized with random values for all methods. The smoothing parameter in the log-square sparsity measure is set to $\nu = 1000$ and the optimal values of κ and μ that balance the sparsity of the solution with the penalty values of the regularizing functions in Eq. (3.17) are determined by a grid search, evaluated on the operators' performance in the reconstruction task described in the subsequent section. All other hyper-parameters of the optimization procedure are chosen heuristically and are set to the same constant values across all experiments. The procedure is stopped when reaching a maximum number of 2000 iterations or when either the norm of the gradient is smaller than $1e-5$ or the total change of analysis operator coefficients is smaller than $1e-4$. The operator learning is executed on an Intel Core i7-3930K CPU with 3.2 GHz and 4GB of memory and completes in approximately four minutes.

Image reconstruction from partial data

The achievable reconstruction quality by the differently learned analysis operators is evaluated on the commonly used test images 'boat', 'house' and 'man'. For each of them, 95% of the pixels in the original image are eliminated to create the measurement vector $\boldsymbol{y}$ by creating the masking operator $\boldsymbol{A}$ which selects the pixels from the images in an independent and identically distributed fashion. Then, all images are reconstructed from the few measurements $\boldsymbol{y}$ by solving the single-channel image recovery problem in Eq. (3.36). As the solver, the conjugate gradient method from [73] is used to minimize the function numerically. The parameter that balances sparsity and data fidelity is set in all runs empirically to $\lambda = 100$ and is shrunk to $\lambda = 1$ subsequently across the iterations. The reconstruction is initialized by a simple linear interpolation of the input pixels in $\boldsymbol{y}$ on the pixel grid. The solver is run for 150 iterations, where λ is decreased after every 50 iterations. Figure 3.6 visualizes input and result of the image reconstruction process.

The final estimate $\boldsymbol{x}^\star$ of each test image is compared to its original version $\boldsymbol{u}$ based on the 8-bit resolution of their gray-scale values, using three common metrics for assessing image quality. The standard metrics for comparing signal quality Root-Mean-Squared Error

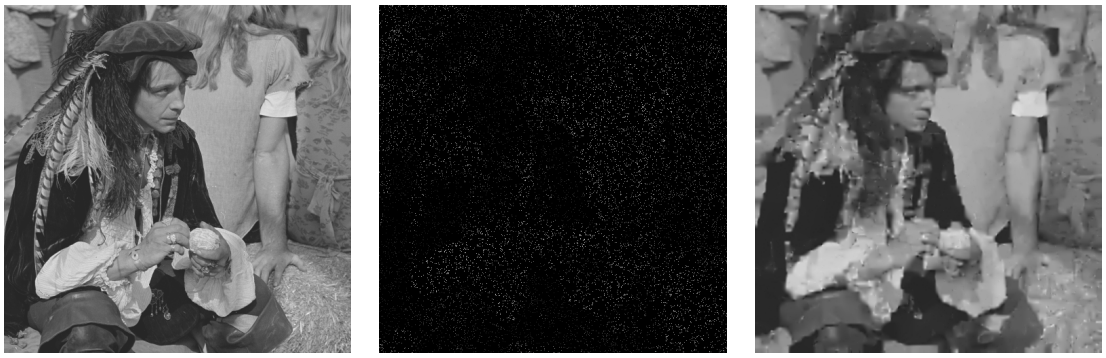


Figure 3.6: Original version (left), 5% of its pixels (middle) that are used as the input to the algorithm and the reconstructed version (right) of the test image 'man'.

(RMSE) and Peak Signal-to-Noise Ratio (PSNR), defined as

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(x_i^\star - u_i)^2}{N}}, \tag{3.37}$$

and

$$\mathrm{PSNR} = 10\log\left(\frac{255^2}{\sum_{i=1}^{N}(x_i^\star - u_i)^2}\right), \tag{3.38}$$

are used. Further, the Structural SIMilarity (SSIM) index from [176] with proposed default parameters is employed to quantify structural similarity of the reconstruction, which better reflects perceptual quality of the images. The results of this quantitative comparison are summarized in Table 3.1.

metric / method	GOAL [74]		GOAL (normalized)		proposed		proposed (reduced rows)	
↑ PSNR	23.490	25.382	23.491	25.342	23.485	**25.428**	**23.497**	25.377
	24.462	24.445	24.399	24.410	**24.473**	**24.462**	24.447	24.440
↑ SSIM	0.759	0.776	**0.763**	0.776	0.762	0.776	0.760	**0.778**
	0.765	0.767	0.765	0.768	**0.768**	**0.769**	0.766	0.768
↓ RMSE	17.062	13.723	17.061	13.787	17.072	**13.650**	**17.049**	13.731
	15.255	15.347	15.368	15.405	**15.237**	**15.319**	15.282	15.354

Table 3.1: Summary of the quantitative comparison of the different learned analysis operators in reconstructing three test images from 5% of their original pixels. Each cell contains the results achieved by the corresponding method on the images in the following order: Top left 'boat', top right 'house', bottom left 'man', and bottom right average over all of them. PSNR is measured in decibels (dB). The best results are printed in bold.

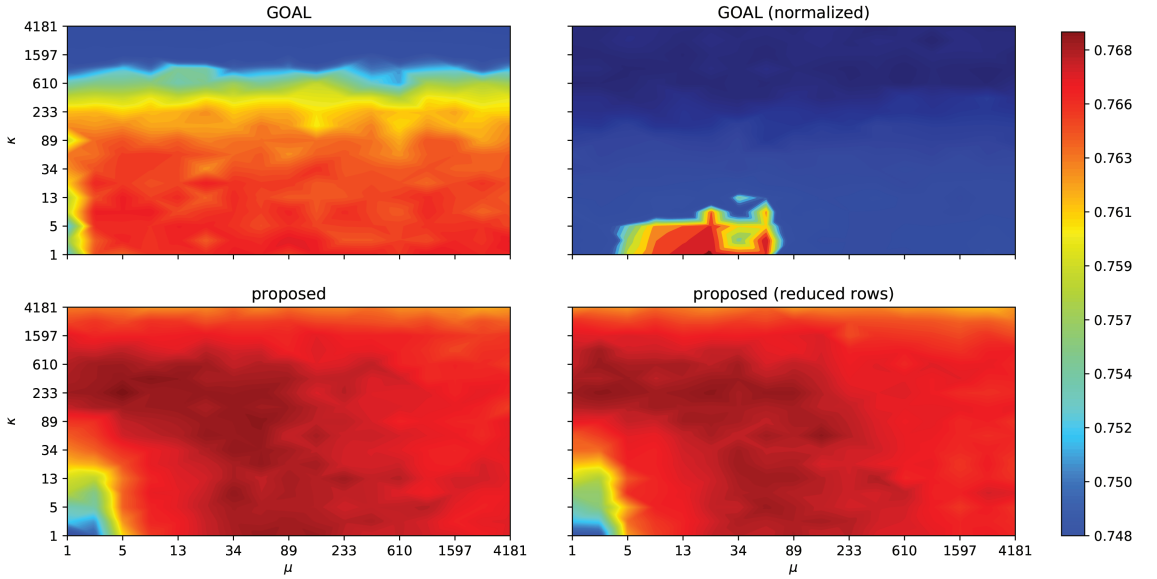


Figure 3.7: Contour plots showing the average SSIM over all test images with respect to the hyper-parameters μ and κ which balance the sparsity objective with the rank and coherence penalties during analysis operator training. Dark red areas represent high (better) and dark blue areas low (worse) average SSIM values.

Besides comparing the image reconstruction quality that can be achieved with each of the trained analysis operators, it is also of interest, how stable the learning methods are when varying the key parameters. For the analysis operator learning, three parameters are most crucial. The first two are the balancing parameters μ and κ in Eq. (3.17), that trade the sparsity of the objective function with the rank and coherence penalties during training. It is desirable to achieve good learning results over a range of parameterization values in order to reliably run the learning algorithm when other aspects of the experiment vary. The sensitivity of the discussed methods with respect to these parameters is analyzed by learning a large number of analysis operators, each time setting a different configuration of μ and κ, while keeping all other parameters of the learning process fixed. Subsequently, the different operators are utilized in the same image reconstruction tasks over the test image set and their reconstruction quality is compared. Figure 3.7 shows contour plots of the achievable reconstruction quality measured by SSIM for varying values of the key parameters μ and κ for the proposed and reference learning methods. As it can be seen, both variants of the proposed method achieve high SSIM scores over a wide range of parameter configurations

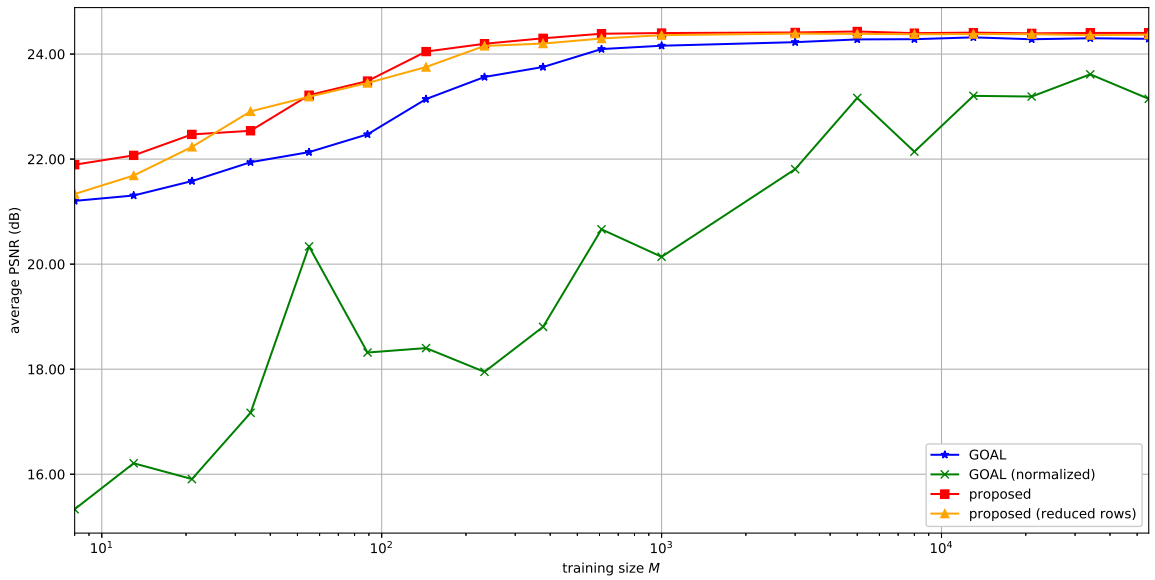


Figure 3.8: Average PSNR of the reconstruction from partial data achieved over all test images with analysis operators trained with different methods and varying size of the training set. The proposed method (red squares) requires fewer training samples to reach a stable output compared to the reference method.

of μ and κ. The results are remarkably more stable in that regard than the ones produced by the reference method.

The third parameter that is of particular interest is the training size M which determines the number of samples that are collected from the training images and are fed to the learning algorithm. It is always desirable for a learning algorithm to require as few training data sample as possible to reliably determine model parameters. For supervised methods, on the one hand, this means less effort in labeling data. For unsupervised methods, such as the presented analysis operator learning, on the other hand, a small number of required training samples leads to a smaller computational effort of the learning procedure which can in consequence be completed in a shorter amount of time. To study the effect of the training size on the learned operators and their ability to regularize well the image reconstruction problem, each of the discussed methods is restarted several times with varying values of the training size M and the resulting learned operators are used in the same image reconstruction problem. Figure 3.8 shows how the training size influences the reconstruction quality for the different methods respectively. It is clear from the line plot that the proposed method

requires fewer training samples to reach stable and high-quality reconstruction results than the reference method.

3.6 Discussion

In this chapter, a model for learning analysis operators for local image structures in photometric data was proposed, which is invariant to variations in brightness and contrast. Further, it was shown that for analysis operators to be illumination invariant, their rows need to be zero-mean. Interestingly, the assumption that structure encoding linear filters are zero-mean is coherent with findings in work of adjacent fields [80, 37]. The proposed model addresses brightness normalized training data by accounts for its particular geometric structure imposed by the normalization procedure. A practical learning algorithm was derived from an existing method by adapting the algorithm in [74] to a manifold structure suitable for brightness normalized data. In an image reconstruction task, where the learned analysis operators are employed as regularizers, it was shown empirically that the proposed model can achieve a higher quality image reconstruction than its reference method. Further, its sensitivity with respect to key parameters of the learning procedure improved significantly. Finally, the proposed model requires less training data to achieve training results reliably. In summary, it could be shown that brightness-invariant learning of local image structures is beneficial when dealing with photometric data and that the resulting reduced data complexity of the learning problem can be exploited with a practical learning algorithm that generates high-quality analysis operators for image reconstruction tasks.

Chapter 4

Co-sparse analysis model for bimodal image data

This chapter is based on the peer-reviewed publications:

M. Kiechle et al. "A Joint Intensity and Depth Co-Sparse Analysis Model for Depth Map Super-Resolution". In: *Proc. International Conference on Computer Vision*. 2013, pp. 1545–1552

M. Kiechle et al. "A Bimodal Co-sparse Analysis Model for Image Processing". In: *International Journal of Computer Vision* 114.2 (2015), pp. 233–247

Processing image data that is captured by a single type of sensor has ubiquitous applications. However, an important trend in designing technical systems lies in the possibility to combine different imaging technologies that record heterogeneous physical properties and consequently sense more information about the environment. Robotics, autonomous driving or remote sensing are popular examples, where the combination of multiple imaging sensor types has advanced the respective fields. Many computer vision tasks can benefit from an ability to exploit interdependency between different image modalities. Fusing information extracted from the different image data modalities can be achieved on several levels, and one promising approach is the integration at a low level.

In this chapter, interdependencies in image data of photometric intensity, scene depth and

near infrared are explored. Building on the centered co-sparse analysis model of the previous chapter, a bimodal co-sparse analysis model that is able to capture the interdependency of two image modalities is proposed. It is based on the assumption that a pair of analysis operators exists, so that the co-supports of the corresponding bimodal image structures have a large overlap. An algorithm is proposed that is able to learn such a coupled pair of operators from registered and noise-free training data. Furthermore, it is demonstrated how this model can be applied to solve linear inverse problems in image processing and how it can be used as a prior in bimodal image registration tasks.

4.1 Multi-modal image data

In the past, the majority of methods tackling problems in computer vision were focused on working on a single image modality, typically a color or grayscale image captured with a digital camera. Due to the progress in sensor technologies, sensors that capture different types of image modalities beyond intensity have become affordable and popular. Well-known examples of multi-modal image sensors combine classical photometric cameras with thermal, multi-spectral and depth cameras. These image signals often carry information about one another and exploiting this interdependency is beneficial for solving various problems in computer vision, such as image reconstruction, registration, segmentation, detection, or recognition, in a more robust way. Inspired by biological systems, which perceive their environment through many different signal modalities at once, fusing sensory information from different modalities has emerged as an important research topic. Existing fusion schemes can be grouped according to their level of fusion. Methods of decision-level fusion work independently on the different modalities to make separate task-dependent decisions, which are then fused according to a certain rule or confidence measure. Feature-level fusion methods integrate modality-specific features to derive a decision, for instance the well-known bag-of-words method in object classification. The method presented here, belongs to the group of low-level fusion, where the multi-modal information is integrated on the pixel level.

Typically, low-level integration is formulated as finding a mapping from the raw pixel measurement domain of one modality to another, such that the transformed values are correlated. In successful approaches, this mapping is learned from sets of aligned local image patches

to make corresponding algorithms computationally tractable [60, 15, 79, 97]. More recent approaches aim at capturing the low-level integration across modalities via sparse coding, where the interdependencies of the signals are reflected in interdependencies of their sparse codes. This concept is used in several methods to find a mapping between different resolution levels or across image modalities. In [186, 191], such a scheme is applied to single image Super-Resolution (SR). Two dictionaries are learned for corresponding Low-Resolution (LR) and High-Resolution (HR) image patches and the two domains are fused through a common sparse representation. In [96], Li et al. propose a SR approach across the two different modalities intensity and depth. Therein, three domains are fused through different dictionaries for LR and HR depth-, as well as HR intensity-patches. The dictionaries are learned under the constraint that the corresponding sparse representation have a common support.

The coupling of different image modalities through the assumption of a common sparse code has turned out to be too strict in practice. In [106], a relaxed joint sparse model is proposed in which a dictionary is learned for the source image domain together with a transformation matrix, which transforms the sparse representations of the source domain to signals in the target domain. Wang et al. [175] use linear regression between the sparse representations over different dictionaries for the image domains. A similar idea is followed by Jia et al. [85], who refine the linear mapping of sparse codes by a local parameter regression for different subsets of sparse representations. While all of these fusion methods rely on the sparse synthesis model, the related co-sparse analysis model [53] has not been considered yet in such a multi-modal setting. This is particularly surprising given its excellent performance in unimodal image processing tasks [121, 73, 185, 37].

In this chapter, a bimodal data model based on co-sparsity for two image modalities is proposed. It allows finding signal representations that have a correlated co-sparse representation across the two different image domains. One advantage of choosing the analysis over the synthesis model is that in the analysis model, the sparse code of a query signal can be obtained extremely fast without the need to solve the sparse coding problem. This enables the use of co-sparsity as a prior in problems such as image registration, which might otherwise be difficult to achieve using synthesis sparsity.

To demonstrate both, the descriptive power and cross-modal coupling of this model, it is first employed as a prior for solving inverse problems, which is validated in an image guided

depth-map reconstruction task. As a second application scenario, the model is used within a novel algorithm for bimodal image registration, which, to the best of the author's knowledge, is the first sparsity-based approach to tackle this problem. Concretely, the proposed joint bimodal co-sparsity model is combined with an optimization on Lie groups which achieves favorable results in comparison to other bimodal image registration methods.

4.2 Bimodal co-sparse analysis model

In this section, the concept of learning co-sparse representations from training signals is combined with the idea of joint sparsity of two different signal domains. Before detailing the approach to this joint co-sparsity model, some important aspects this method is built upon need to be reviewed.

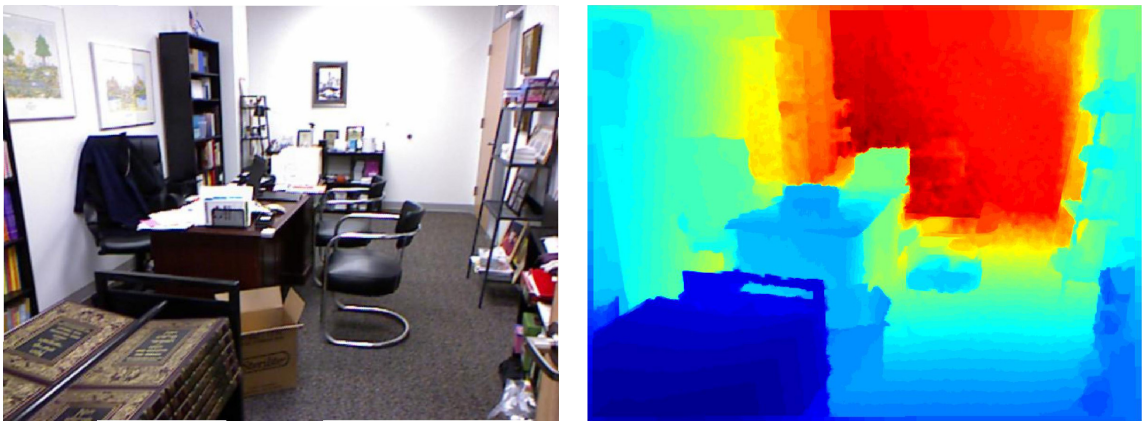
Joint analysis co-sparsity

In this approach, two signal classes X_U and X_V of different modalities are considered that emanate from the same physical object. Consider for example an intensity image and a depth map captured from the same scene as depicted in Figure 4.1. More precisely, let $(\boldsymbol{x}_U, \boldsymbol{x}_V) \in \mathrm{X}_U \times \mathrm{X}_V$. It may be assumed that these signal pairs $(\boldsymbol{x}_U, \boldsymbol{x}_V)$ allow a co-sparse representation with an appropriate pair of analysis operators $(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) \in \mathbb{R}^{l \times n_U} \times \mathbb{R}^{l \times n_V}$.

Based on the knowledge that the structure of a signal is encoded in its co-support, it may be further assumed that *a pair of analysis operators exists such that the intersection of the co-supports of* $\boldsymbol{\Omega}_U \boldsymbol{x}_U$ *and* $\boldsymbol{\Omega}_V \boldsymbol{x}_V$ *is large.* Thus, let

$$\left[\boldsymbol{\Omega}_U \boldsymbol{x}_U, \boldsymbol{\Omega}_V \boldsymbol{x}_V\right] =: \begin{bmatrix} \boldsymbol{a}_1^\top \\ \vdots \\ \boldsymbol{a}_l^\top \end{bmatrix} = \boldsymbol{A} \in \mathbb{R}^{l \times 2},$$

the concatenated sparse codes, then the proposed bimodal co-sparse analysis model is based on the idea that $\boldsymbol{A}$ will be row-wise sparse. This is typically achieved by minimizing a mixed



(a) photometric intensity image (RGB) **(b)** structured light depth image

Figure 4.1: Exemplary pair of spatially registered color intensity and scene depth map from the NYU Depth Dataset [124].

ℓ_2-ℓ_0-norm [167, 118, 82], formally denoted by

$$\|\boldsymbol{A}\|_{2,0} = \left\| \begin{bmatrix} \|\boldsymbol{a}_1\|_2 \\ \vdots \\ \|\boldsymbol{a}_l\|_2 \end{bmatrix} \right\|_0 . \tag{4.1}$$

Geometrically interpreted, the goal is to partition the signal space for each of the two modalities in such a way, that the partitions not only represent subsets of unimodal signals but simultaneously relate to a partition of the other modality.

Specifically, the objective is to learn the coupled pair of bimodal analysis operators $(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) \in \mathbb{R}^{l \times n_U} \times \mathbb{R}^{l \times n_V}$ for two signal modalities. Therefore, a set of M aligned and corresponding training pairs

$$\{(\boldsymbol{x}_U^{(i)}, \boldsymbol{x}_V^{(i)})\}_{i=1}^{M} \subset \mathbb{R}^{n_U} \times \mathbb{R}^{n_V} \tag{4.2}$$

is used. For simplicity, the training signals and their analysis operators of both modalities are assumed to be of the same size throughout this chapter, i.e. $n_U{=}n_V{=}n$, although it is easily verified that this is conceptually not required.

As a consequence, this means that the sought solution $(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V)$ minimizes the sum or

empirical mean of $\|\boldsymbol{A}\|_{2,0}$ over all sample pairs within the training set. Analogously to previous chapters, it is paramount to relax the non-smooth ℓ_0-sparsity term. Here, the log-square sparsity measure from Eq. (3.11) is applied to the joint representation $\boldsymbol{A}$ of $\boldsymbol{x}_U$ and $\boldsymbol{x}_V$, which reads as

$$\begin{aligned} g\colon \mathbb{R}^{l\times 2} \to \mathbb{R}\colon \boldsymbol{A} \mapsto g(\boldsymbol{A}) &:= \frac{1}{\log(1+\nu)} \sum_{j=1}^{l} \log\left(1+\nu\|\boldsymbol{a}_j\|^2\right) \\ &= \frac{1}{\log(1+\nu)} \sum_{j=1}^{l} \log\left(1+\nu\big((\boldsymbol{\Omega}_U\boldsymbol{x}_U)_j^2 + (\boldsymbol{\Omega}_V\boldsymbol{x}_V)_j^2\big)\right) \\ &= \frac{1}{\log(1+\nu)} \sum_{j=1}^{l} \log\left(1+\nu\big(\boldsymbol{z}_{U,j}^2 + \boldsymbol{z}_{V,j}^2\big)\right). \end{aligned} \tag{4.3}$$

This sparsity function is used throughout this chapter to measure and control joint co-sparsity of a pair of bimodal analysis operators and local image patches.

Since the ideal pair of bimodal operators represents all samples in the training set, the sum of empirical variance σ_g^2 and squared mean of the sparsity measure $\bar{g}^2$ from Eq. (4.3) is minimized over all training signal pairs

$$\begin{aligned} f(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) &:= \bar{g}^2 + \sigma_g^2 \\ &= \frac{1}{M} \sum_{i=1}^{M} g(\boldsymbol{\Omega}_U\boldsymbol{x}_U^{(i)}, \boldsymbol{\Omega}_V\boldsymbol{x}_V^{(i)})^2. \end{aligned} \tag{4.4}$$

The derivation is a straight-forward adaptation from Eq. (3.16).

Regularization of rank and coherence

As with all existing co-sparse analysis operator learning approaches (see Chapter 2), controlling rank and coherence of the operator rows is important for the bimodal operator pair as well. Unlike the sparsity objective, which couples the representation learning across the two signal modalities, the regularizers need to be applied to the two operators individually, since neither of the transformations should yield trivial information about its signal domain. Ac-

cordingly, the rank and coherence constraints are modeled through their respective penalty functions $h(\boldsymbol{\Omega})$ from Eq. (3.9) and $r(\boldsymbol{\Omega})$ from Eq. (3.10) applied to both analysis operators individually

$$p(\boldsymbol{\Omega}) := \kappa h(\boldsymbol{\Omega}) + \mu r(\boldsymbol{\Omega}). \tag{4.5}$$

Like before, κ and μ determine the impact of the corresponding regularization on one operator. Combining the ingredients of joint co-sparsity and operator regularization, the smooth learning function is written as

$$L\left(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V\right) := f(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) + p(\boldsymbol{\Omega}_U) + p(\boldsymbol{\Omega}_V). \tag{4.6}$$

Choice of the manifold

Before the bimodal analysis operator pair can be learned from data by minimizing the learning objective in Eq. (4.6), the scale-orthogonality ambiguity of the sparsity problem needs to be addressed. As in the previous chapter, this is solved here by applying the norm constraint from Eq. (3.7), which fixes the scale of operator rows to a constant. Accordingly, the minimization of $L\left(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V\right)$ is carried out on an appropriate manifold. The choices are the oblique manifold as in [74] or the manifold of unit norm and zero mean signals $\mathcal{R}$ from the definition in Eq. (3.8), which makes the training invariant to the mean of training signals. Depending on the type of image data, one or the other may be more appropriate. Section 4.4.4 contains an empirical study with both settings. For notation brevity, $\mathcal{R}$ will be used to refer to both of them subsequently.

Finally, the learning function for jointly co-sparse analysis operators is obtained by combining the individual operator regularizers with the joint co-sparsity objective and is stated as

$$(\boldsymbol{\Omega}_U^\star, \boldsymbol{\Omega}_V^\star) \in \underset{\boldsymbol{\Omega}_U^\top, \boldsymbol{\Omega}_V^\top \in \mathcal{R}}{\arg\min}\ L(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V). \tag{4.7}$$

4.3 Joint bimodal analysis operator learning algorithm

In order to obtain an optimal solution to Eq. (4.7), the conjugate gradient method on matrix manifolds, outlined in Appendix A, is employed. In comparison to the learning with only a single modality, in each iteration of the optimization procedure both analysis operators $\boldsymbol{\Omega}_U$ and $\boldsymbol{\Omega}_V$ need to be updated. The geometric CG framework supports this setting without modifications if the formulation considers the two operators in combination as a product structure where the objective function acts on elements of it. In this setting, the gradient formulation needs adjustment. Concretely

$$\nabla L(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) = [\nabla_U L(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V), \nabla_V L(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V)], \tag{4.8}$$

where ∇_U and ∇_V denote the gradient of L with respect to its first and second input. The derivatives with respect to U and V are decoupled for penalty terms, resulting in

$$\begin{aligned} \nabla_U L(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) &= \nabla_U f(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) + \nabla_U p(\boldsymbol{\Omega}_U) \\ \nabla_V L(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) &= \nabla_V f(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) + \nabla_V p(\boldsymbol{\Omega}_V). \end{aligned} \tag{4.9}$$

The derivatives for $p(\boldsymbol{\Omega})$ are given in Eq. (3.29) and Eq. (3.30). Finally, the derivatives of the joint co-sparsity measure are required. Using the notation of the last line of Eq. (4.3), they read as

$$\nabla_U f(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) = \frac{2}{M} \frac{1}{\log(1+\nu)^2} \left[\sum_{i=1,j=1}^{M,l} \frac{2\nu z_{U,ji} \left(\sum_{j=1}^{l} \log(1 + \nu(z_{U,ji}^2 + z_{V,ji}^2)) \right)^2}{1 + \nu(z_{U,ji}^2 + z_{V,ji}^2)} \boldsymbol{J}_{ji} \right] \boldsymbol{X}^\top \tag{4.10}$$

and

$$\nabla_V f(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V) = \frac{2}{M} \frac{1}{\log(1+\nu)^2} \left[\sum_{i=1,j=1}^{M,l} \frac{2\nu z_{V,ji} \left(\sum_{j=1}^{l} \log(1 + \nu(z_{U,ji}^2 + z_{V,ji}^2)) \right)^2}{1 + \nu(z_{U,ji}^2 + z_{V,ji}^2)} \boldsymbol{J}_{ji} \right] \boldsymbol{X}^\top. \tag{4.11}$$

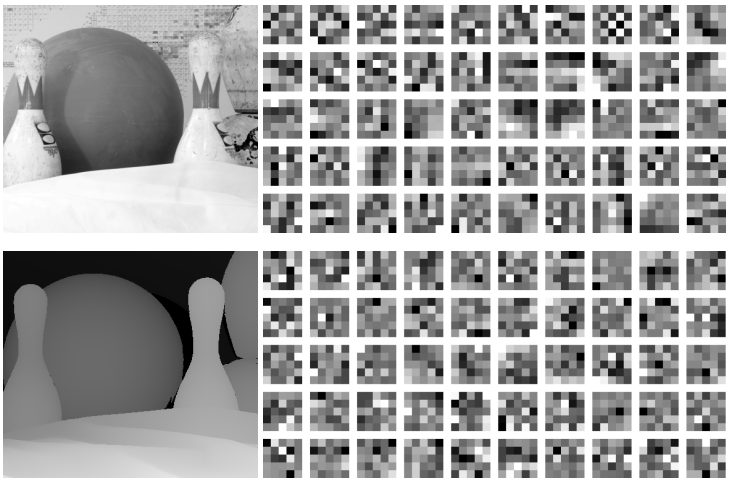


Figure 4.2: Example pair of input training images (left) and learned rows of bimodal operator pairs (right) visualized as square patches for intensity (top) and depth (bottom). Each square patch corresponds to a row of the learned operators $\boldsymbol{\Omega}_U$ and $\boldsymbol{\Omega}_V$ (top left patch corresponds to first operator row respectively).

Equipped with the Euclidean gradient $\nabla L \in \mathbb{R}^{l \times (n_U + n_V)}$ of the learning objective, the geometric CG can then be executed on the manifold $\mathcal{R} \times \mathcal{R}$.

One important assumption in this model is that the training signals of each pair are spatially registered such that corresponding pixel locations in the two images refer to observations of the same point in the environment. Concerning the choice of training samples, M pairs of square patches are randomly sampled from aligned, noise-free images and each patch is vectorized to form the i-th training vector pair $(\boldsymbol{x}_U^{(i)}, \boldsymbol{x}_V^{(i)})$ and the analysis operators are initialized with random values before starting the optimization procedure. Learning such pairs of operators from several thousand samples on a standard desktop PC can be accomplished within the order of a few minutes. For the experiments in Section 4.4 and Section 4.5, two operators are trained from $M = 15000$ patch pairs with a size of $\sqrt{n} = 5$. The algorithm completes this task on an Intel Core i7 3.2 GHz CPU with 4GB of memory and unoptimized Matlab code in about 140s. Figure 4.2 illustrates rows of learned operator pairs as square patches for the two bimodal image setups intensity and depth.

4.4 Bimodal image reconstruction

To evaluate the usefulness of the proposed bimodal image model, it is employed as a prior in a highly ill-posed image reconstruction task formulated as an inverse problem. Specifically, the task involves reconstructing an image to high quality from a low-resolution input image with noise and missing values by providing a high-resolution second image that is spatially aligned but only available in a different image modality. This setup has real world applications in many sensor fusion settings. RGB-D cameras are a popular example that combine a high-resolution digital camera with an additional low-resolution scene depth sensor. Using the proposed method, a bimodal analysis operator is learned on clean training data and then applied as a prior to regularize the reconstruction problem.

4.4.1 Formulation of the bimodal image reconstruction problem

The general goal of the bimodal image reconstruction task is to recover an aligned pair of bimodal images $\boldsymbol{x}_U, \boldsymbol{x}_V \in \mathbb{R}^N$ from a set of measurements $\boldsymbol{y}_U \in \mathbb{R}^{m_U}, \boldsymbol{y}_V \in \mathbb{R}^{m_V}$. Here, $\boldsymbol{x}_U, \boldsymbol{x}_V$ are vectorized versions of the original images from each of the two modalities, obtained by ordering their entries lexicographically. Without loss of generality, it is assumed that the images $\boldsymbol{x}_U, \boldsymbol{x}_V$ are of the same size and number of pixels N.

In the reconstruction approach, the problem of bimodal image reconstruction is posed as a linear inverse problem. Formally, the relation between $\boldsymbol{x}_U, \boldsymbol{x}_V$ and $\boldsymbol{y}_U, \boldsymbol{y}_V$ is given by

$$\boldsymbol{y}_U = \boldsymbol{A}_U \boldsymbol{x}_U + \boldsymbol{n}_U, \qquad \boldsymbol{y}_V = \boldsymbol{A}_V \boldsymbol{x}_V + \boldsymbol{n}_V. \tag{4.12}$$

$\boldsymbol{A}_U \in \mathbb{R}^{m_U \times N}, \boldsymbol{A}_V \in \mathbb{R}^{m_V \times N}$ model the sampling process of the measurements and $\boldsymbol{n}_U \in \mathbb{R}^{m_U}, \boldsymbol{n}_V \in \mathbb{R}^{m_V}$ model noise and potential sampling errors. For typical reconstruction tasks, the dimensions m_U, m_V of the measurement vectors may be significantly smaller than the dimension N. Consequently, reconstructing $\boldsymbol{x}_U, \boldsymbol{x}_V$ in Eq. (4.12) is highly ill-posed.

To resolve this, the bimodal data model is employed as a co-sparsity prior to regularize the image reconstruction. To apply the co-sparsity prior learned from image patches to the full extent of the image, the global version of the analysis operator $\boldsymbol{\Omega}^F$, as defined in Section 3.4,

is used. Accordingly, one aims to solve

$$(\boldsymbol{x}_U^\star, \boldsymbol{x}_V^\star) \in \underset{\boldsymbol{x}_U, \boldsymbol{x}_V \in \mathbb{R}^N}{\arg\min}\, d_E\left((\boldsymbol{A}_U \boldsymbol{x}_U, \boldsymbol{A}_V \boldsymbol{x}_V), (\boldsymbol{y}_U, \boldsymbol{y}_V)\right) + \lambda g(\boldsymbol{\Omega}_U^F \boldsymbol{x}_U, \boldsymbol{\Omega}_V^F \boldsymbol{x}_V). \tag{4.13}$$

d_E denotes an appropriate data error measure and the weighting factor λ is used to balance the solution between data fidelity and joint co-sparsity prior. Due to the joint co-sparsity term, the analyzed versions of both modalities are enforced to have a correlated co-support and as a result, the two signals are coupled.

The reconstruction task formulation in Eq. (4.13) is general and various application configurations are possible. First, depending on the choice of the measurement operators $\boldsymbol{A}_U, \boldsymbol{A}_V$, different reconstruction types such as denoising, inpainting, or upsampling can be performed. Second, the reconstruction can be accomplished jointly on both signals simultaneously or only on one single modality, while the other serves as a guiding reference to the co-sparsity and data priors. In the following section it is demonstrated, how image guided depth map super-resolution can be accomplished using this setup.

4.4.2 Image-guided depth map reconstruction

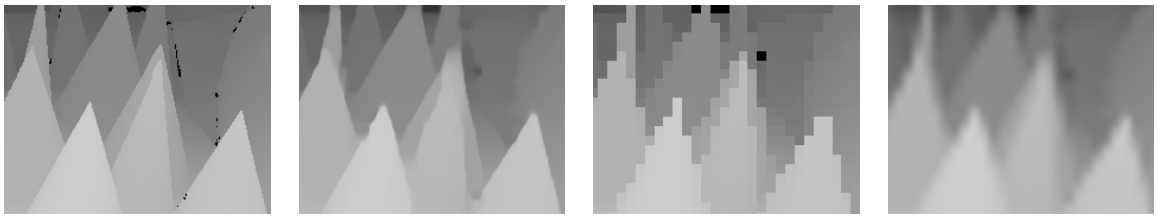
In this experiment, the proposed reconstruction approach is applied to the image modalities intensity and depth. Due to the availability of affordable sensors, this has become a common bimodal image setup. The focus lies now on recovering the HR depth image $\boldsymbol{x}_D$ from LR depth measurements $\boldsymbol{y}_D$, given a fixed, high quality intensity image $\boldsymbol{x}_I$. In this case, $\boldsymbol{A}_I$ is the identity operator, implying $\boldsymbol{y}_I = \boldsymbol{x}_I$ and the analyzed intensity image is constant, i.e.

$$\boldsymbol{\Omega}_I^F \boldsymbol{x}_I = \mathbf{c} = const. \tag{4.14}$$

This simplifies Eq. (4.13) for recovering the HR depth map to

$$\boldsymbol{x}_D^\star \in \underset{\boldsymbol{x}_D \in \mathbb{R}^N}{\arg\min}\, d_E(\boldsymbol{A}_D \boldsymbol{x}_D, \boldsymbol{y}_D) + \lambda g(\boldsymbol{c}, \boldsymbol{\Omega}_D^F \boldsymbol{x}_D). \tag{4.15}$$

The choice of the data fidelity term d_E depends on the error model of the depth data and can be adjusted to a specific setup. For instance, this may be a measure tailored to a sensor



(a) original **(b)** proposed method **(c)** NN interp. **(d)** bicubic interp.

Figure 4.3: A detail of the depth image 'Cones' (a) from [154] after it was downsampled by a factor of 8 in vertical and horizontal direction and subsequently upscaled using the proposed method (b) and standard interpolation methods (c, d).

specific error model. Subsection 4.4.5 includes an example with a specific noise model for the Microsoft Kinect sensor. The fixed HR intensity image implicitly guides the reconstruction of the depth image due to its coupling in the co-sparsity prior. Thusly, information about the scene gained from the intensity image and its co-support regarding the bimodal analysis operators helps to regularize the HR depth signal.

4.4.3 Prior art on depth map reconstruction

Increasing the resolution of depth images obtained from range sensors has become an important research topic, and diverse approaches treating this problem have been proposed throughout the past years. Many of these methods originate from the closely related problem of intensity image super-resolution. However, these mostly aim at producing pleasantly looking results, which is different from the goal of achieving geometrically sound depth maps. Straightforward upsampling methods like nearest-neighbor, bilinear, or bicubic interpolation produce undesirable staircasing or blurring artifacts, as can be seen in Figure 4.3. In the following, many well-established methods for depth map SR that aim at reducing these artifacts are reviewed briefly.

Initially, methods were proposed that use smoothing priors from edge statistics [57] or local self-similarities [59]. These methods only require a single image, but either have difficulties in textured areas, or only work well for small upscaling factors. A different approach, which also solely requires depth information is based on fusing multiple displaced LR depth maps into a single HR depth map. Schuon et al. [155] develop a global energy optimization

framework employing data fidelity and geometry priors. This idea is extended for better edge-preservation by Bhavsar et al. in [19].

A number of recently introduced methods aim at exploiting co-aligned discontinuities in intensity and depth images of the same scene. They fuse the HR and LR data utilizing Markov Random Fields (MRF). In the work of Diebel and Thrun [47], color image information is used to guide depth reconstruction by computing the smoothness term in Markov-Random-Field formulation according to texture derivatives, which is extended in [100] by a data term better adapted to depth images, and combined with depth from passive stereo in [192]. In order to better preserve local structures and to remove outliers, Park et al. [138] add a non-local means term to their MRF formulation. Mac Aodha et al. [102] treat depth SR as a MRF labeling problem of matching LR depth map patches to HR patches from a predefined database.

Inspired by successful stereo matching algorithms, Kopf et al. [93] and Yang et al. [187] apply bilateral filtering to depth cost volumes in order to iteratively refine an estimate using an additional color image. Chan et al. [33] elaborate on this approach with a fast and noise-aware joint bilateral filter. Xiang et al. [181] include sub-pixel accuracy, and Dolson et al. [48] address temporal coherence across a depth data stream from Light Detection and Ranging (LiDaR) scanners by combining a bilateral filter with a Gaussian framework.

Finally, methods exist that exploit the dependency between sparse representations of intensity and depth signals over appropriate dictionaries. In [67], the complex wavelet transform is used as the dictionary. Both the HR intensity image and the LR depth map are transformed into this domain and the resulting coefficients are fused using a dual tree to obtain the HR depth map. Instead of using predefined bases, approaches employing learned dictionaries are known to lead to state-of-the-art performance in diverse classical image reconstruction tasks, cf. [55, 105]. Surprisingly, applying those techniques for depth map enhancement has only been explored very recently. Mahmoudi and Sapiro [104] first learn a depth dictionary from noisy samples, then refine and denoise these samples and finally learn an additional dictionary from the denoised samples to inpaint, denoise, and super-resolve projected depth maps from 3D models. Closest to the proposed approach are the recent efforts of [96] and [165]. They independently learn dictionaries of depth and intensity samples, and model a coupling of the two signal types during the reconstruction phase. In [96], three dictionar-

ies are composed from LR depth, HR depth, and HR color samples to learn a respective mapping function based on edge features. In contrast, only two dictionaries for intensity and depth are learned in [165], where the similarity of the support of corresponding sparse representations is used to model the coupling.

4.4.4 Evaluation on stereo data

To compare the results of the proposed approach to state-of-the-art methods, the algorithm is quantitatively evaluated on the four standard test images 'Tsukuba', 'Venus', 'Teddy', and 'Cones' from the Middlebury stereo dataset [154]. The required LR input depth maps are created artificially by downscaling the ground truth depth maps by a factor of d in both vertical and horizontal dimension. Before downsampling, the available HR image is blurred with a Gaussian kernel of size $(2d-1) \times (2d-1)$ and standard deviation $\sigma = d/3$. The LR depth map and the corresponding HR intensity image are provided as input to the proposed algorithm.

In this reconstruction from LR measurements, the measurement error is assumed to be independent and identically distributed. An appropriate data fidelity measure for such error distribution is the squared Euclidean distance

$$d_E(\boldsymbol{A}_D \boldsymbol{x}_D, \boldsymbol{y}_D) = \|\boldsymbol{A}_D \boldsymbol{x}_D - \boldsymbol{y}_D\|_2^2. \tag{4.16}$$

Plugging this term into Eq. (4.15) yields the formulation for image guided super-resolution of the HR depth image

$$\boldsymbol{x}_D^\star \in \underset{\boldsymbol{x}_D \in \mathbb{R}^N}{\arg\min} \|\boldsymbol{A}_D \boldsymbol{x}_D - \boldsymbol{y}_D\|_2^2 + \lambda g(\boldsymbol{c}, \boldsymbol{\Omega}_D^F \boldsymbol{x}_D). \tag{4.17}$$

This problem is solved using a standard conjugate gradient method and an Armijo step size selection. To achieve the best results within few iterations, the parameter λ is set to a large value initially and the conjugate gradient optimization procedure is restarted several times, while consecutively shrinking the multiplier to a final value of $\lambda = 1$. The problem in Eq. (4.17) is not convex and convergence to a global minimum can not be

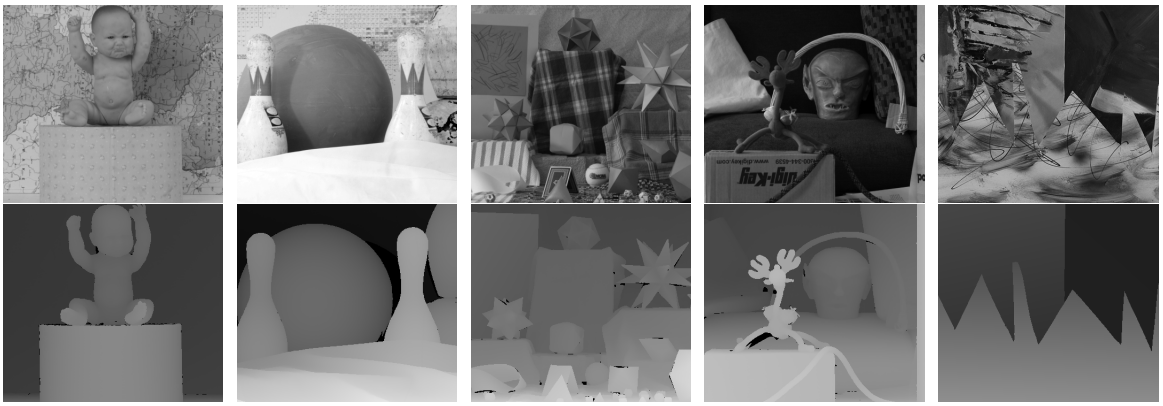


Figure 4.4: The five training images from [154] used for learning the bimodal intensity-depth analysis operator. The intensity images (top) and corresponding depth maps (bottom) are spatially registered prior to sampling patch pairs for training.

guaranteed. In practice, however, accurate depth maps are always obtained from different random initializations of $\boldsymbol{x}_D$.

For this evaluation, an operator pair is trained once offline and used in all following intensity and depth experiments. To that end, a total of $M = 15000$ pairs of square sample patches of size $\sqrt{n} = 5$ are gathered from the five registered intensity and depth image pairs 'Baby1', 'Bowling1', 'Moebius', 'Reindeer' and 'Sawtooth', taken from the Middlebury stereo set [154]. The image pairs are depicted in Figure 4.4 for reference. Furthermore, the operators are learned with twofold redundancy, i.e. $l = 2n$, resulting in the operator pair $(\boldsymbol{\Omega}_I, \boldsymbol{\Omega}_D) \in \mathbb{R}^{50\times 25} \times \mathbb{R}^{50\times 25}$. The remaining learning parameters are set empirically to $\nu = 400, \kappa_I = 5, \kappa_D = 22, \mu_I = 10^2$ and $\mu_D = 2.5 \cdot 10^4$. The learning is conducted with two different manifold settings, OB indicating learning on the oblique manifold and $\mathcal{R}$ on the set of rows that are unit norm and zero mean.

Following the methodology described in the work of other depth map SR approaches, the Middlebury stereo matching online evaluation tool[1] is used to quantitatively assess the accuracy of produced results with respect to the ground truth data. The percentage of bad pixels over all pixels in the depth map with an error threshold of $\delta = 1$ is reported along with the RMSE based on 8-bit pixel value resolution. The results are compared to several

[1] http://vision.middlebury.edu/stereo/eval/

d	method	Tsukuba	Venus	Teddy	Cones
2x	nearest-neighbor	1.24	0.37	4.97	2.51
	Yang et al. [187]	1.16	0.25	2.43	2.39*
	Diebel and Thrun [47]	2.51	0.57	2.78	3.55
	Hawe et al. [73]	1.03	0.22	2.95	3.56
	proposed (OB)	**0.47**	**0.09**	**1.41**	**1.81**
	proposed ($\mathcal{R}$)	0.83*	0.12*	1.96*	2.69
4x	nearest-neighbor	3.53	0.81	6.71	5.44
	Yang et al.	2.56	0.42	5.95	4.76*
	Diebel and Thrun	5.12	1.24	8.33	7.52
	Hawe et al.	2.95	0.65	4.80	6.54
	proposed (OB)	1.73*	0.25*	**3.54**	5.16
	proposed ($\mathcal{R}$)	**1.48**	**0.23**	3.99*	**4.69**
8x	nearest-neighbor	3.56	1.90	10.9	10.4
	Yang et al.	6.95	1.19	11.50	11.00
	Diebel and Thrun	9.68	2.69	14.5	14.4
	Lu et al. [100]	5.09	1.00	9.87	11.30
	Hawe et al.	5.59	1.24	11.40	12.30
	proposed (OB)	3.53*	**0.33**	**6.49**	9.22*
	proposed ($\mathcal{R}$)	**3.30**	0.34*	8.11*	**8.57**

Table 4.1: Numerical comparison of the proposed method to other depth map SR approaches for different upscaling factors d. The figures represent the percentage of bad pixels with respect to all pixels of the ground truth data and an error threshold of $\delta = 1$. The best and second best results are highlighted in bold and with an asterisk respectively. OB and $\mathcal{R}$ indicate learning of the analysis operators with the two different manifold settings.

state-of-the-art methods for image guided depth map SR [47, 187, 33, 100, 74] and the metrics are listed in Table 4.1 and Table 4.2.

The proposed method improves depth map SR considerably over simple interpolation approaches as depicted in Figure 4.3. Neither staircasing nor substantial blurring artifacts occur, particularly in areas with discontinuities. Also, there is no noticeable texture cross-talk in areas of smooth depth and cluttered intensity. Edges can be preserved with great detail due to the additional knowledge provided by the intensity image, even if SR is conducted using large upscaling factors. The quantitative comparison with other depth map SR methods demonstrates the excellent performance of the presented approach.

d	method	Tsukuba	Venus	Teddy	Cones
2x	nearest-neighbor	0.612	0.288	1.543	1.531
	Chan et al. [33]		0.216	1.023	1.353
	Mac Aodha et al. [102]	0.601	0.296	0.977	1.227
	Hawe et al. [74]	0.278	0.105	0.996	0.939
	proposed (OB)	**0.255**	**0.075**	**0.702**	**0.680**
	proposed ($\mathcal{R}$)	0.256*	0.077*	0.803*	0.821*
4x	nearest-neighbor	1.189	0.408	1.943	2.470
	Chan et al.		0.273	**1.125**	1.450
	Mac Aodha et al.	0.833	0.395	1.184*	1.779
	Hawe et al.	0.450*	0.179	1.389	1.398
	proposed (OB)	0.487	0.129*	1.347	1.383*
	proposed ($\mathcal{R}$)	**0.374**	**0.108**	1.256	**1.287**
8x	nearest-neighbor	1.135	0.546	2.614	3.260
	Chan et al.		0.369	**1.410**	**1.635**
	Hawe et al.	0.713*	0.249	1.743	1.883
	proposed (OB)	0.753	0.156*	1.662	1.871*
	proposed ($\mathcal{R}$)	**0.660**	**0.155**	1.729*	1.931

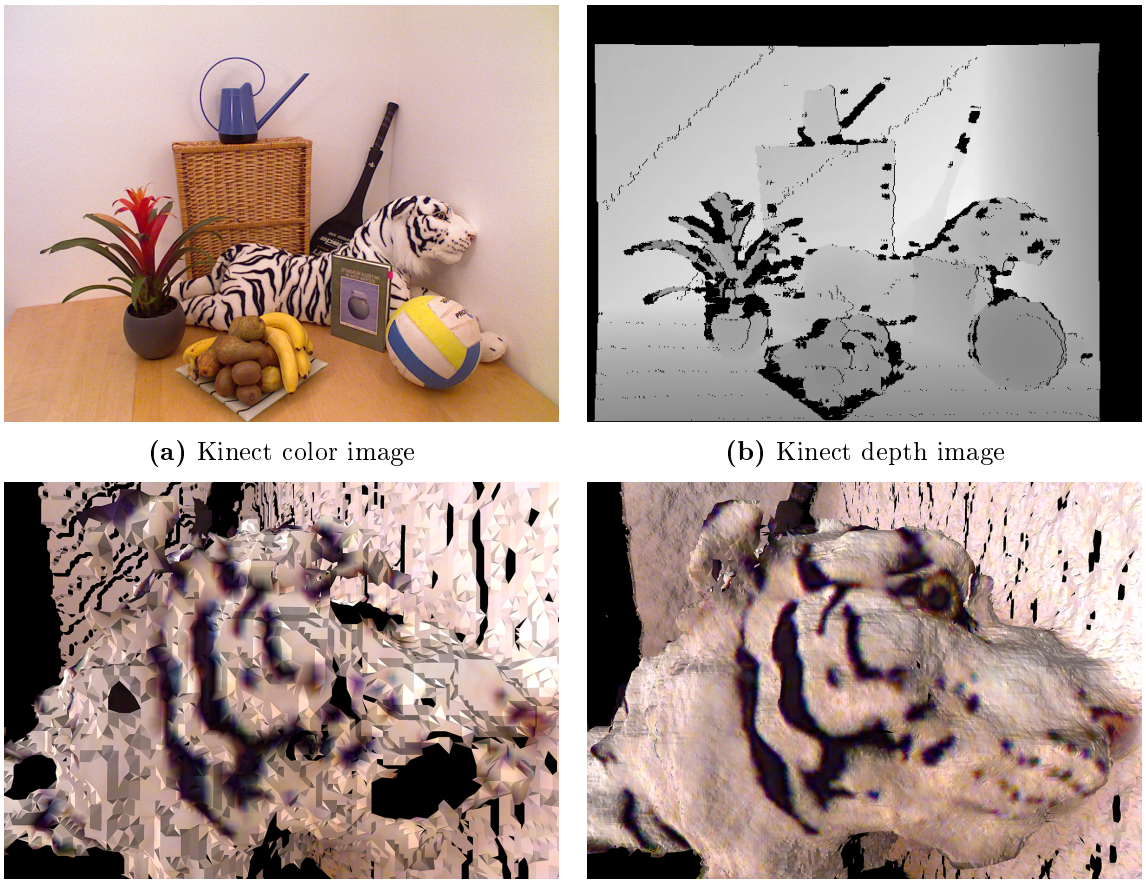
Table 4.2: Numerical comparison of the proposed method to other depth map SR approaches. The figures represent the RMSE in comparison with the ground truth depth map. The best and second best results are highlighted in bold and with an asterisk respectively. OB and $\mathcal{R}$ indicate learning of the analysis operators with the two different manifold settings.

4.4.5 Validation on Kinect data

In order to demonstrate the applicability of the proposed algorithm to real data, another experiment is conducted, where color images of size 1280x960 and corresponding depth maps of size 640x480 are captured using the Microsoft Kinect sensor. The depth maps are then upscaled by a factor of $d = 2$ to match their size to the color images.

Since the approximate error statistics for this application and this sensor were studied previously in [89], one can use this information to further refine the data model. According to [89], the standard deviation of Kinect depth data is proportional to the square of the depth value $\sigma_i \propto (y_D^{(i)})^2$. This may be utilized in the error model by employing the squared Mahalanobis distance for d_E in Eq. (4.15), which yields

$$\boldsymbol{x}_D^\star \in \underset{\boldsymbol{x}_D \in \mathbb{R}^N}{\arg\min} \left(\boldsymbol{A}_D \boldsymbol{x}_D - \boldsymbol{y}_D\right)^\top \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{A}_D \boldsymbol{x}_D - \boldsymbol{y}_D\right) + \lambda g(\mathbf{c}, \boldsymbol{\Omega}_D^F \boldsymbol{x}_D) \tag{4.18}$$



(a) Kinect color image

(b) Kinect depth image

(c) without refinement

(d) after refinement

Figure 4.5: Color image (a) and corresponding registered depth map (b) recorded by the Kinect sensor as well as 3D renderings of the tiger head detail visualizing the difference between the original sensor data (c) and the refined version using the proposed method (d).

where $\boldsymbol{\Sigma} \in \mathbb{R}^{m_2 \times m_2}$ is a diagonal matrix with main diagonal elements $(y_D^{(i)})^2$.

As the Kinect sensor uses structured light to measure depth, the signal is corrupted by missing pixels due to occlusions arising from the displacement of the infrared light source and the sensor. To fill these gaps in the data, the measurement matrix is modeled in such a way that it excludes these gaps from the sampling process of the LR depth image, i.e. removing the rows of $\boldsymbol{A}$ that correspond to zero entries in $\boldsymbol{y}_D$. As a result, the reconstruction algorithm performs inpainting of missing depth values without any additional processing,

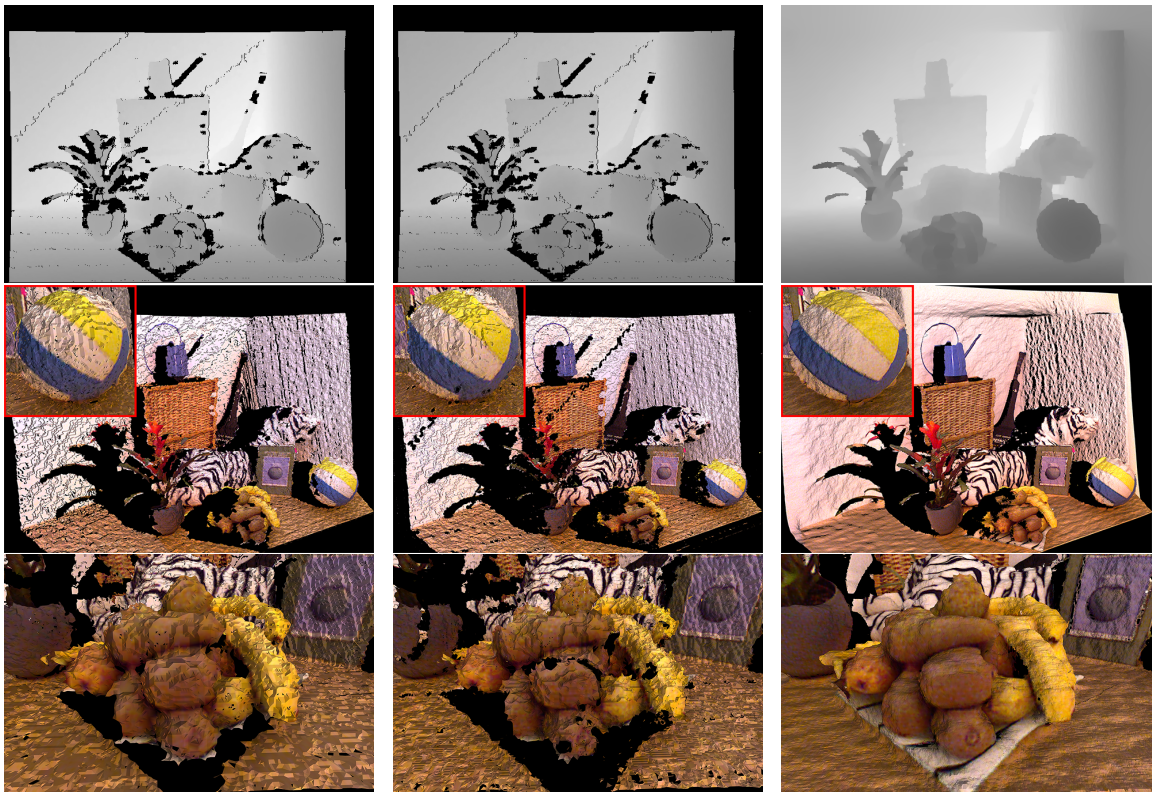


Figure 4.6: Depth maps (top row), 3D rendering of Kinect color and depth data depicting the entire scene (middle row) and a detail of the fruit bowl (bottom row). Left column: original Kinect data like in the top row of Figure 4.5 with downsampled color information, center column: bicubic interpolation (1280x960), right column: proposed method (1280x960). Note that object shadows are due to the single view occlusions.

while simultaneously increasing the depth map resolution. In this way, two of the main issues of Kinect data are handled in a single step.

To the author's knowledge, there is no data set publicly available that allows to numerically evaluate Kinect depth map enhancing methods by providing ground truth data. Therefore, the quality of the super-resolved Kinect depth maps is assessed qualitatively. Since small differences in the depth map represented as a gray-scale image are almost invisible to the naked eye, the results are illustrated in Figure 4.6 using ball pivoting surface reconstruction [17] on a point cloud that was created from the depth map computed by the proposed algorithm. As it can be seen, the method does not only increase the details in the 3D scene significantly, but also treats the missing pixels with great success. This is especially obvious

in the details of the tiger head in Figure 4.5 and the fruit bowl in Figure 4.6. The 3D rendering illustrates the impact of the bimodal support during reconstruction particularly around depth discontinuities, but it also leads to smoother surfaces of table and wall due to the smooth texture of the corresponding intensity signal. It needs to be emphasized that for this experiment, the same analysis operators are used as in in the Middlebury stereo data experiment in Section 4.4.2. The prior seems to work well on Kinect data, even though the training data to learn the analysis operators was captured using a different sensor technology than the Kinect. This underpins that the learned prior model is general enough to be used for high quality reconstruction of both synthetic and real world data.

4.5 Bimodal image registration

During the training of bimodal analysis operators, one important assumption made in Section 4.3, is the fact that training data is spatially aligned. The learned analysis operator pair therefore models local image structure in corresponding image modalities, given that they are spatially aligned. In the bimodal image reconstruction task, spatial alignment of the two images was guaranteed, while the model was used to reconstruct local image structure. This section describes an approach to solve the opposite problem: given two images from different modalities whose internal structure is uncorrupted, can the spatial alignment be restored with the help of the learned data model?

4.5.1 Prior art on bimodal rigid image alignment

Image registration is the process of geometrically aligning two images that were taken by e.g. different sensors, at different points in time or from different viewpoints. Automatic image registration can be categorized into feature-based and area-based algorithms. The first group of algorithms searches for salient features in both images (e.g. edges, corners, contours) and tries to find the matching pairs of features. The geometric transformation that minimizes the distance between matching features is then used to transform one of the images. Area-based algorithms do not consider specific features but use the whole overlapping region between both images to evaluate the registration. In both cases a distance metric is needed to either

match the features or to measure the similarity between image regions. In the unimodal registration case simple metrics like the sum of squared differences or correlation can be used. Multimodal registration is more challenging because the intensities of two different sensors can differ substantially when imaging the same physical object. This phenomenon is often called contrast reversal, as bright objects in one modality can be very dark in the other and vice versa. In general, no straight-forward functional relationship between the intensities of the sensors exists. Nevertheless, the approach of Orchard [134] tries to find a piecewise linear mapping between the intensities of different modalities. The most popular metric for multimodal image registration is Mutual Information, originally introduced by Viola and Wells [174] and Collignon et al. [40]. A normalized version was later proposed by Studholme et al. [161] that is better suited to changing sizes of the overlapping region. Mutual Information is used for a variety of different applications and sensors as in medical registration [142], remote sensing [39, 56] and surveillance [94]. More information and an extensive review of further image registration approaches covering several decades of research in this area is covered in [24] and [193].

4.5.2 Bimodal image registration algorithm

In the following, an area-based approach that employs the formerly learned bimodal co-sparse analysis model for registration of two image modalities is presented. Two images I_U and I_V of a 3D scene are considered that are sensed through two modalities U and V. Further, it is assumed that these images can be aligned with a transformation τ that belongs to one of the following Lie groups $\mathcal{G}$.

- The special orthogonal group $SO(2)$;
- the special Euclidean group $SE(2)$;
- the special affine group $SA(2)$;
- or, the affine group $A(2)$.

This means that, if $\boldsymbol{p}$ denotes the homogeneous pixel coordinates for one modality, say I_U, there exists some $\tau \in \mathcal{G}$ such that the two images are perfectly aligned

$$I_V(\tau \boldsymbol{p}) \sim I_U(\boldsymbol{p}) \quad \text{for all pixel coordinates } \boldsymbol{p}. \tag{4.19}$$

Here, the standard representation of the above groups in the set of (3×3) real matrices was chosen, and the standard group action $\tau \boldsymbol{p}$ on the homogeneous coordinates is simply given by a matrix-vector multiplication. Note, that the inclusions $SO(2) \subset SE(2) \subset SA(2) \subset A(2)$ hold. The shorthand notation $\tau \circ I$ will be used for the transformed image, i.e.

$$(\tau \circ I) := I(\tau \boldsymbol{p}) \quad \text{for all pixel coordinates } \boldsymbol{p} \tag{4.20}$$

and cubic interpolation is applied to calculate the pixel values. The goal of this section is to find τ by using the bimodal pair of analysis operators $(\boldsymbol{\Omega}_U, \boldsymbol{\Omega}_V)$. The idea behind this approach is, that for an optimal transformation, the coupled sparsity measure should be minimized. Thus, one is searching for $\tau^\star \in \mathcal{G}$ such that

$$\tau^\star \in \operatorname*{arg\,min}_{\tau \in \mathcal{G}} g\Big(\boldsymbol{\Omega}_U^F I_U, \boldsymbol{\Omega}_V^F (\tau \circ I_V)\Big). \tag{4.21}$$

In order to tackle the above optimization problem, an approach that is similar to what has been proposed in [141] is followed. It is based on iteratively updating the estimate of τ with group elements near the identity. Locally, the matrix exponential yields a diffeomorphism between a neighborhood of the identity in $\mathcal{G}$ and a neighborhood around 0 in the corresponding Lie algebra $\mathfrak{g}$ of $\mathcal{G}$. For the considered Lie groups at hand, each Lie algebra is contained in

$$\mathfrak{g} := \left\{ \begin{bmatrix} \boldsymbol{A} & \boldsymbol{b} \\ 0 & 0 \end{bmatrix} \mid \boldsymbol{A} \in \mathbb{R}^{2\times 2}, \boldsymbol{b} \in \mathbb{R}^2 \right\}, \tag{4.22}$$

which is the Lie algebra of $A(2)$. Further restrictions on the parameters then lead to the corresponding Lie algebras for the respective sub groups: $\boldsymbol{A}^\top = -\boldsymbol{A}$ and $\boldsymbol{b} = 0$ for $SO(2)$, $\boldsymbol{A}^\top = -\boldsymbol{A}$ for $SE(2)$, and $\operatorname{tr} \boldsymbol{A} = 0$ for $SA(2)$.

Thus, for a transformation δ which is near the identity, we have $\delta = e^{\boldsymbol{H}}$ for some matrix $\boldsymbol{H} \in \mathfrak{g}$ in a neighborhood of 0. In order to tackle the optimization problem of Eq. (4.21),

the following procedure is followed. For legibility, denote

$$F(\tau) := g\Big(\boldsymbol{\Omega}_U^F I_U, \boldsymbol{\Omega}_V^F(\tau \circ I_V)\Big). \tag{4.23}$$

The geometric gradient descent method outlined in Appendix A is applied on the Lie group $\mathcal{G}$ for minimizing $F(\tau)$ that updates τ in each step. To that end, the set of (3×3) real matrices is endowed with the inner product

$$\langle \boldsymbol{H}_1, \boldsymbol{H}_2 \rangle_{\boldsymbol{P}} := \operatorname{tr}\left((\boldsymbol{H}_1 \odot \boldsymbol{P})\boldsymbol{H}_2^\top\right), \tag{4.24}$$

with $\boldsymbol{P}$ having positive entries and $\odot$ denoting the Hadamard product. The choice of $\boldsymbol{P}$ allows balancing the translational versus the rotational part of the chosen group, or the shearing part, respectively. This is commonly done to account for different magnitudes of the transformation parameters [92]. The algorithm below outlines the optimization procedure. Here, the Armijo rule is chosen to determine the step size.

Choose the Lie group $\mathcal{G}$ of admissible transformations and set $\tau_0 := \mathrm{id}$ (identity) as an initialization. Then iterate the following steps until convergence.

1. Compute the Riemannian gradient of $F(\delta \circ \tau)$ at $\delta = \mathrm{id}$, which is an element of the Lie algebra

$$\boldsymbol{G} := \operatorname{grad}_\delta F(\delta \circ \tau)\big|_{\delta=\mathrm{id}} \in \mathfrak{g}. \tag{4.25}$$

2. Use the Armijo rule to choose $t^\star$

$$t^\star \approx \arg\min F(e^{t\boldsymbol{G}} \circ \tau). \tag{4.26}$$

3. Update $\tau \leftarrow e^{t^\star \boldsymbol{G}}\tau$.

The derivation of the Riemannian gradient of $F(\delta \circ \tau)$ is provided in the Appendix B.1. As a stopping criterion, a threshold for the norm of the Riemannian gradient is chosen empirically.

4.5.3 Evaluation

The proposed image registration method is compared to two multimodal registration metrics, namely Mutial Information (MI) and Normalized Mutial Information (NMI) [114]. The elastix image registration toolbox [92] provides the reference implementations of these metrics together with a gradient descent algorithm to find the transformation parameters. For all methods, the default parameters of the elastix toolbox are used. In these experiments, the intensity and depth images from the Middlebury stereo set and images from the RGB-NIR Scene Dataset [25] are used for evaluation. The RGB-NIR dataset consists of RGB images and Near-Infrared (NIR) images that were captured with commercial Digital Single Lens Reflex (DSLR) cameras using filters for the visible and infrared spectrum. The spectra do not overlap (the cutoff wavelength is about 750 nm) and the NIR images give statistically different information from the R, G and B channel. Both datasets are very well aligned and this registration is used as the ground truth for learning the operators on aligned training images.

One fixed operator pair is trained for each of the registration scenarios intensity+depth and intensity+NIR. For the intensity and depth setup, the same operator is used as in the reconstruction experiments in Section 4.4.2. For the intensity and NIR setup, the same learning procedure is followed, randomly collecting $M = 15000$ pairs of square sample patches of size $\sqrt{n} = 5$ from a total of 9 images in the training set, one from each category which is subsequently excluded from testing. The learning parameters are empirically set to $\nu = 200, \kappa_I = 250, \kappa_N = 1000, \mu_I = 250$ and $\mu_N = 1000$. All other parameters are the same as for intensity and depth.

In order to evaluate the result of the registration of one image pair, a synthetic deregistration is applied to one of the images. This deregistration consists of a translation and a rotation and subsequently the registration algorithm searches for a transformation that belongs to the special Euclidian group. Both the elastix toolbox and the proposed algorithm work on a Gaussian image pyramid of four levels.

Table 4.3 shows the remaining registration error after running the different registration algorithms. The proposed method achieves comparable or better results than MI or NMI for all the modalities. The MI and NMI algorithms fail to register the intensity and depth image

deregistration (x,y,θ)	method	intensity-depth (art)	intensity-NIR (old building)	intensity-NIR (mountain)
0, 0, 10	MI	14.70, 50.14, -2.01	6.14, 1.56, -3.01	**0.16**, **-0.85**, **-3.52**
	NMI	9.42, 18.66, -7.83	2.49, 4.26, -9.17	-1.43, -1.05, -3.85
	proposed	**-1.11**, **-1.41**, **0.11**	**0.35**, **0.22**, **0.01**	-0.34, -2.95, -8.31
-5, -5, 0	MI	-1.06, **1.13**, 0.02	-0.21, -0.14, 0.05	0.10, -0.06, **0.05**
	NMI	7.02, 4.96, **0.01**	**0.03**, **0.10**, **0.02**	**0.05**, **0.01**, 0.06
	proposed	**-1.00**, -1.13, 0.03	-1.03, 2.34, 2.72	2.56, 0.41, 0.06
10, 0, 5	MI	8.60, 18.71, -2.49	-8.59, 2.26, -1.32	2.64, 0.08, -1.76
	NMI	3.44, 9.79, -3.27	-8.22, 2.27, -1.35	2.58, **0.05**, -1.71
	proposed	**-0.90**, **-0.81**, **0.19**	**0.02**, **0.23**, **0.06**	**0.61**, 0.37, **-0.02**
-5, -5, 5	MI	1.38, 11.12, -2.65	-7.53, 1.79, -1.43	1.57, -0.23, -1.77
	NMI	3.04, 8.87, -3.01	-8.83, 1.86, -1.36	1.58, -0.27, -1.73
	proposed	**-0.22**, **-1.40**, **0.28**	**0.22**, **-0.13**, **0.14**	**0.82**, **-0.01**, **-0.28**

Table 4.3: Registration residual for different synthetic translations and rotations. Values for the translation in x and y direction are given in pixels, the angle θ is given in degrees. The best results are printed bold.

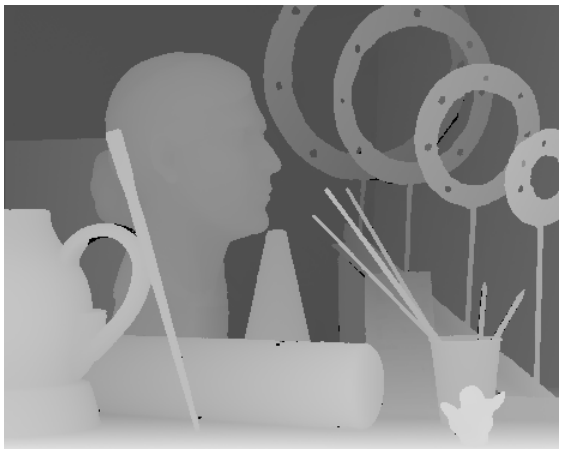
pair in most of the cases. This can be explained by the fact that intensity and depth are much less alike than intensity and near-infrared (see Figure 4.7) and the proposed algorithm can benefit from the learned operator pair that is adapted to the respective scenarios. The MI and NMI algorithms do not require this learning stage and are therefore better suited to tasks where the modalities are very similar.

Figure 4.8 shows the registration error for various initial deregistrations of the intensity and depth image pair. Here, the remaining combined registration error is defined as

$$\epsilon = \sqrt{\epsilon_x^2 + \epsilon_y^2 + \epsilon_\theta^2}, \tag{4.27}$$

with ϵ_x and ϵ_y denoting the remaining translation error in x- and y-direction (in pixels) and ϵ_θ denoting the remaining rotation error (in degrees). Dark blue areas in Figure 4.8 correspond to small registration errors ($\epsilon < 1$) and red areas show large errors ($\epsilon > 50$), i.e. configurations where the registration failed. It can be seen that MI is susceptible to large translations and fails to align the images correctly. The direct comparison of the proposed method and NMI shows that both algorithms can handle the initial deregistration better than MI but the proposed method achieves smaller remaining errors over a wider range of deregistration values.

(a) Art (intensity)



(b) Art (depth)

(c) Old building (intensity)

(d) Old building (NIR)

(e) Mountain (intensity)

(f) Mountain (NIR)

Figure 4.7: Example images used in the registration experiments. The intensity and depth image pair differ significantly, which is challenging for multimodal registration algorithms. The NIR images are more similar to the intensity images and differ mainly in areas with vegetation and sky.

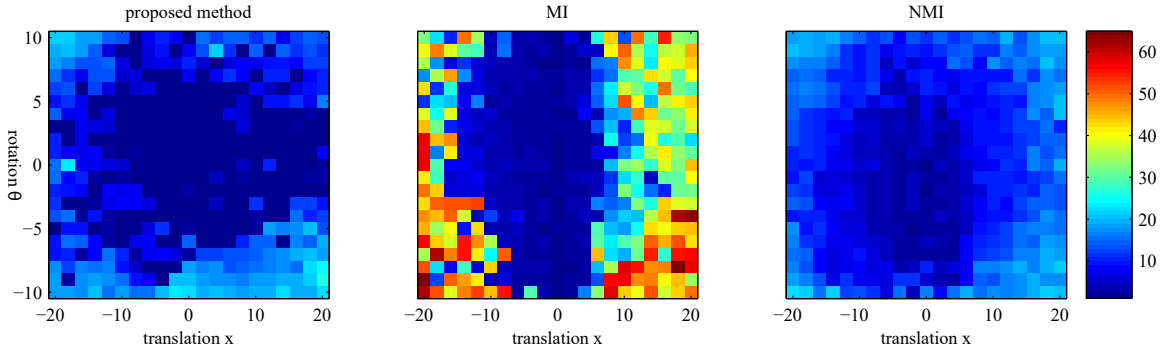


Figure 4.8: Remaining combined registration error for different initial deregistrations which consist of a translation in x-direction and a rotation θ. Dark blue and red areas correspond to small and large errors, respectively. MI fails to register the images for large translations. The proposed method achieves the smallest remaining error and can handle large translations and rotations.

Figure 4.9 shows how the proposed method performs for registration of an intensity and depth image pair using an affine transformation.

(a) misaligned intensity-depth image pair **(b)** after alignment

Figure 4.9: Example of an intensity and depth image pair before (a) and after (b) the proposed registration process using an affine transformation.

4.6 Discussion

In this chapter, modeling the interdependencies of two image modalities by applying the analysis model in a joint co-sparsity setup was proposed. The coupled analysis operators were learned by minimizing a joint co-sparsity function via a conjugate gradient method on an appropriate manifold.

The descriptive power of the learned model was evaluated in two different application scenarios. First, it was used as a regularizer for the inverse problem of bimodal image reconstruction and numerical experiments for image-guided depth map super-resolution were provided. As a second application scenario, the problem of bimodal image registration was considered. An algorithm on Lie groups was proposed that uses an afore learned pair of bimodal analysis operators to register intensity and depth images as well as intensity and NIR images.

The experiments in both applications show that the proposed model is indeed a very useful tool in bimodal image modeling. Nevertheless, some observations regarding applicability and limitations of the approach need a discussion. The model is based on the assumption that a pair of analysis operators exists such that analyzed bimodal image patches have co-supports with significant overlap. In practice, one interesting case is the occurrence of constant patches in one image modality. Since constant patches always yield maximal co-support, they trivially fit the model. This has two notable consequences.

First, when learning the model parameters, constant patches do not influence the value of the cost function due to the pre-processing (constant patches are set to zero) and thus the learning of the operators. Considering a toy-example where one modality is always constant, this would lead to learning a unimodal analysis operator only for the other modality.

The second consequence concerns image analysis in terms of using the joint sparsity measure as a prior. Here, a patch pair fits the model best, if it achieves the largest joint co-sparsity. In this regard, given a patch in one modality, a constant patch in the other modality will always be most suitable. This results in an advantage and a drawback. On one hand, forcing joint co-sparsity if one patch is constant is equivalent to forcing sparsity only in the other modality, which leads to the unimodal co-sparsity model. Considering that constant patches are often caused by a phenomenon that is simply not observable in both image modalities at

once (photometric texture on a smooth surface for instance), this is clearly an advantage. On the other hand, forcing joint co-sparsity if one patch is not constant will lead to a constant patch in the other modality without any further constraints. This is the reason why task-specific constraints, e.g. data fitting, are important. This limitation, however, is not specific to this bimodal model per se, but also valid for the unimodal co-sparse analysis model.

Chapter 5

Co-sparse analysis model for unsupervised texture segmentation

This chapter is based on the peer-reviewed publication:

M. Kiechle et al. "Model-Based Learning of Local Image Features for Unsupervised Texture Segmentation". In: *IEEE Transactions on Image Processing* 27.4 (Apr. 2018), pp. 1994–2007

So far, co-sparse analysis operator learning has been discussed as an approach to learn robust low-level image models from training data and the previous chapters have shown how they can be applied in image reconstruction tasks with great success. In this chapter, the local image features that are learned by the analysis operator are explored in their application to another well-known problem in image processing and computer vision: image segmentation. Here, the goal is to partition the 2D image plane into sections that represent distinct objects in the 3D environment. Since co-sparse analysis features excel in describing local image structures, it is tempting to test their usefulness in image segmentation. Based on the assumption that the label map of a useful image segmentation is typically piecewise constant in the image plane, the question is whether a subset of analysis features can be selected automatically, such that their descriptors of local structure are approximately piecewise constant and are therefore good candidates for image segmentation features.

5.1 Texture segmentation

Texture segmentation is a frequently occurring and challenging problem in image processing and computer vision. For textured images – such as many natural images [113, 119], histological images [115], or crystal structures [117] – the segmentation is typically performed in two stages. In the first stage, a (vector-valued) feature image is derived from the image. The corresponding features are designed to capture the local statistical properties or oscillatory patterns of a texture. Many classical features are based on linear filters [143], for example Gabor filters [83], wavelet frames [170], windowed Fourier transform [14], followed by a pointwise non-linearity [171]. Other popular features are based on local spectral histograms [98], morphological filters [180], local statistical descriptors [164] or local binary patterns [130]. In the second stage, the feature image is segmented. Popular choices include k-means clustering [83, 170] or mean shift algorithms [135]. More sophisticated (variational) segmentation models additionally enforce spatial regularity of the segment boundaries: here, a prominent example is the piecewise constant Mumford-Shah model (or Potts model) [62, 120], which has been used for texture segmentation, for instance in [147, 87, 160, 117].

Prior work on unsupervised texture segmentation

Besides the aforementioned works, there is a series of more recent contributions to unsupervised texture segmentation: Todorovic and Ahuja create a tessellation of texture superpixels (texels) and cluster them by a multiscale segmentation and a meanshift algorithm [164]. Galun et al. [61] utilize a multiscale aggregation of filter responses and shape elements. Haindl and Mikeš employ a Gaussian Markov Random Field (GMRF) texture model [71] or a 3D auto regressive model [68], and they perform segmentation based on a Gaussian mixture model. Scarpa et al. [153] use features based on Markov chains, and then segment by recursively merging them according to their mutual interaction. Yuan et al. [189] use local spectral histograms as feature vectors and formulate the segmentation problem as a multivariate linear regression. In a follow-up work [188], non-negative matrix factorization is used for segmentation. Storath et al. [160] utilize monogenic curvelets as features and perform segmentation based on the piecewise constant Mumford-Shah model. In contrast to this work, the features in [160] are computed from a fixed system of handcrafted filters which

are not learned. The method of Panagiotakis et al. [136, 137] is based on voting of blocks, Bayesian flooding and region merging. Mevenkamp and Berkels [117] use local Fourier features, which are tailored to images with crystal structures, and segment using a convex relaxation of the piecewise constant Mumford-Shah model. McCann et al. [115] utilize features derived from local histograms, and segment using non-negative matrix factorization and image deconvolution.

It is a fundamental issue that the performance of the features depends strongly on the class of images or even on the single image. For instance, good features for a natural image may perform poorly on a histological image. Even more, good features for one natural image may not perform as well on another natural image. Thus, the design of the features is a critical task and there are several approaches to this. A straightforward idea is to simply increase the number of features hoping that at least some features are well suited for the texture patterns of the processed image. Unfortunately, the computational effort for segmenting large feature spaces is very high in practice, in particular for segmentation methods which enforce regularity of the boundaries. To circumvent these problems, a commonly used strategy is to manually select a subset from the aforementioned larger set of features, see for example [87, 188]. However, the manual selection requires human supervision which typically results in an expensive, time-consuming task. In principle, for each new class of images one should reevaluate this selection. To avoid manual design of features for each image or each class of images, it seems natural to learn them from data. In a supervised learning setup, where a sufficiently large training set of images with similar characteristics and a ground truth segmentation is available, one can use generic methods, for example the super-pixelation based method of [146] or more recent methods based on convolutional neural networks [99]. In the present unsupervised setup, such a training set is not available. As a consequence, the challenge is to find a suitable objective function for the learning task and a practical numerical procedure to optimize the features accordingly.

In this chapter a method for unsupervised texture segmentation is developed where the features are learned from non-annotated data, i.e. from images without ground truth segmentation. The main contributions of this work are *(i)* a model for feature learning of image features for texture segmentation in the absence of annotated training data, and *(ii)* a practical algorithm for unsupervised texture segmentation based on that model.

Regarding the basic model *(i)*, the starting point is the observation that features are often designed such that the feature image is approximately constant on a texture segment. This allows utilizing segmentation algorithms based on a local homogeneity assumption. The basic idea of this model is to utilize analysis operators to model local image structure of textures in a way that they produce approximately piecewise constant feature images. In other words, analysis operator rows are learned such that the codes they produce for neighboring pixels are roughly equal. As formalized in Section 3.4, the inner products of an analysis operator row with all patches centered around each pixel of an image is equivalent to a two-dimensional convolution. Therefore, the resulting codes will be referred to as convolutional features in this chapter. Besides reasonable constraints on the filters, such as their norm and mutual coherence, the objective is to minimize the cost function of the popular piecewise constant Mumford-Shah segmentation model, i.e. the total length of the discontinuity set of the corresponding feature image.

Regarding *(ii)*, learning filters based on the proposed model turns out to be a challenging optimization problem because it involves a non-smooth and non-convex cost function on the (non-convex) unit sphere. To make it computationally tractable, this model is decomposed and relaxed, resulting in two stages of filter learning and of segmentation. For the relaxed learning stage, a smooth (yet non-convex) approximation of the cost function is employed. To minimize this cost function, the geometric conjugate gradient descent method described in Appendix A is adapted such that it fits with the proposed model. For the segmentation stage, the Lagrange formulation of the piecewise constant Mumford-Shah model is employed. In particular, implied by the model, a data term based on the Mahalanobis distance is considered. To solve the corresponding problem, the approach proposed in [159, 158] is extended in order to be able to deal with the Mahalanobis distance. Finally, a post-processing as in [188] merges small spurious regions to large ones.

The proposed method is evaluated on different types of textured images. A standard benchmark for texture-based segmentation is the Prague texture segmentation benchmark [70]. Here, the proposed method achieves a top rank. In particular, the proposed method gives significantly better results than many earlier methods [164, 61, 71, 68, 153, 189], and slightly better results than the more recent methods proposed in [188, 117]. Further, the proposed method is competitive with the currently leading method Priority Multi Class Flooding

Algorithm (PMCFA) [136, 137]. Besides, the proposed approach provides satisfactory segmentation results on the data set of histological images of [115]. It is emphasized that, although this is a quite different image class, only minor adjustments to the algorithm parameterization are necessary. This shows in particular the flexibility of the proposed method, and the potential for segmenting quite different classes of textured images.

5.2 A model for unsupervised filter learning for texture segmentation

As mentioned in the beginning of this chapter, the goal here is to learn suitable features for texture segmentation when no training data with ground truth is available. The focus lies again on learning convolutional features inspired by the analysis operators of the previous chapters. Convolutional filters are a natural choice because they describe the class of linear translation-invariant filters. A feature image is created by applying linear filtering followed by a (pointwise) nonlinear transform. More precisely, given an image $\boldsymbol{U} \in \mathbb{R}^{h \times w}$, K different convolution filters $\boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_K$ are considered and the resulting filtered images, given in Matlab-type notation by

$$\mathcal{F}_{:,:,1} = \boldsymbol{\Phi}_1 \boldsymbol{U}, \ \ldots, \ \mathcal{F}_{:,:,K} = \boldsymbol{\Phi}_K \boldsymbol{U}. \tag{5.1}$$

In short-hand notation, they read as $\mathcal{F} = \boldsymbol{\Phi}\boldsymbol{U}$. Then, to each filter response, the same nonlinear transformation σ is applied pixel wise. In general, σ is chosen to be symmetric, i.e. $\sigma(x) = \sigma(-x)$. Further, it is required that it has fast decaying slope for large x in order to be robust towards outliers in the filter responses. The nonlinear transform has proven to be beneficial for texture segmentation: according to [171], its purpose is to translate differences in dispersion characteristics into differences in mean value. For further details on choosing σ, see [171]. Here, a logarithmic non-linearity of the form $\sigma(x) := \log(1 + \mu x^2)$ with the free parameter $\mu > 0$ is proposed. The nonlinear transform is considered to be fixed, and one is interested in finding suitable linear convolution operators $\boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_K$, which define the features

$$\mathcal{V} = \sigma(\boldsymbol{\Phi}\boldsymbol{U}). \tag{5.2}$$

Here, $\mathcal{V}$ and $\boldsymbol{\Phi U}$ are three dimensional arrays in $\mathbb{R}^{h\times w\times K}$, and σ has to be understood as componentwise application of its scalar version.

Since the unsupervised setup is considered, no labeled training data (i.e. no ground truth segmentation) for learning the $\boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_K$ is available. In particular, there is no straightforward way to devise a loss function for the learning process. It is proposed to utilize a loss function based on the segmentation model, which in this case is the piecewise constant Mumford-Shah or Potts model: ideally, the features $\mathcal{V}$ are approximately constant on the texture, and the segment boundaries are sufficiently regular. The idea is to learn suitable filters $\boldsymbol{\Phi}$ in a way such that their responses (after applying the non-linearity) on the segments are approximately constant. Minimizing the cost function defined by the length of the discontinuity set of $\mathcal{V}$, denoted by $\|\nabla\mathcal{V}\|_0$ is proposed. More precisely, as a model for choosing the convolution kernels $\boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_K$,

$$\min_{\mathcal{V},\boldsymbol{\Phi}} \|\nabla\mathcal{V}\|_0 \quad \text{subject to} \quad d(\mathcal{V}, \sigma(\boldsymbol{\Phi U})) \leq \varepsilon, \tag{5.3}$$

is suggested with $\varepsilon > 0$. Here, the minimum is taken with respect to both $\boldsymbol{\Phi}, \mathcal{V}$, where the $\boldsymbol{\Phi}_k$ have unit length, zero mean, and fulfill an incoherence and a certain center condition. (Section 5.3.4 elaborates on these constraints). The symbol d denotes a metric, in this case the Mahalanobis distance as explained in Section 5.4. It is noted, that an optimal pair $\boldsymbol{\Phi}^*, \mathcal{V}^*$ of Eq. (5.3) already consists of an optimal filter bank $\boldsymbol{\Phi}^*$ together with a corresponding optimal segmentation $\mathcal{V}^*$.

The model in Eq. (5.3) is computationally hard to access. In particular, the simultaneous optimization w.r.t. both $\boldsymbol{\Phi}$ and $\mathcal{V}$ is extremely demanding. As an approximative strategy, a two stage approach is designed as follows. As a first step, the filters $\boldsymbol{\Phi}$ are optimized using a relaxation of Eq. (5.3) as described in Section 5.3. For the second step, it is noticed, that for fixed $\boldsymbol{\Phi}$, the Lagrange form of Eq. (5.3) is the piecewise constant Mumford-Shah model. Therefore, performing a piecewise constant Mumford-Shah segmentation w.r.t. the Mahalanobis distance (described in Section 5.4) for the obtained feature image is most suitable. It should be pointed out that even this second step of solving the piecewise constant Mumford-Shah problem is known to be an Non-deterministic polynomial-time (NP)-hard problem on its own.

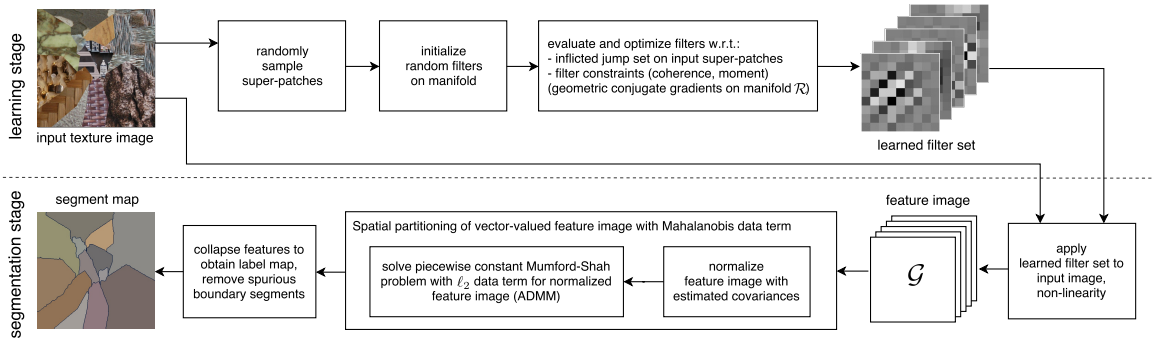


Figure 5.1: Conceptual schematic of the proposed method with its learning and segmentation stages.

Figure 5.1 illustrates the conceptual flow of the proposed method with its filter learning and segmentation stages. For notational brevity, the derivation of the method is described on the basis of gray-valued images $\boldsymbol{U} \in \mathbb{R}^{h \times w}$. The derivation for multi-channel images follows the same basic steps. The relevant modifications regarding the operators $\boldsymbol{\Phi}$ and the jump penalty are described in Section 5.3.6.

5.3 Learning stage

In this section, it is discussed how to learn the filters $\boldsymbol{\Phi}$ from a given image. As a first step, a near anisotropic discretization of the jump penalty in Eq. (5.3) is presented in Section 5.3.1. Then, the model in Eq. (5.3) is relaxed to obtain a computationally better accessible surrogate problem to perform the learning task in Section 5.3.2. Further, learning from patch samples is incorporated in Section 5.3.3, and it is explained how to deal with the constraints imposed on the filters in Section 5.3.4, respectively. Next, the simplified learning problem is summarized and its numerics are discussed in Section 5.3.5. Finally, an explanation how to generalize the approach for multi-channel images is provided in Section 5.3.6.

As mentioned, the focus in this work lies on sets of linear filters. Further, it is assumed that each filter has a fixed number of n coefficients $\boldsymbol{\Phi}_k \in \mathbb{R}^n$.

5.3.1 Near isotropic discretization

First, one has to deal with a near isotropic discretization of the jump penalty $\|\nabla\mathcal{V}\|_0$. As in [158], a finite difference discretization of the form

$$\|\nabla\mathcal{V}\|_0 = \sum_{s=1}^{S} \omega_s \|\nabla_{a_s}\mathcal{V}\|_0 \tag{5.4}$$

is used. The vectors $a_s \in \mathbb{Z}^2 \setminus \{0\}$ belong to a finite difference system $\mathcal{N}$ with $S \geq 2$ elements. For $a \in \mathbb{Z}^2$, let

$$\|\nabla_a\mathcal{V}\|_0 = |\{i = (i_1, i_2) : \|\mathcal{V}_{i,:} - \mathcal{V}_{i+a,:}\|_2 \neq 0\}|, \tag{5.5}$$

where the notation $\mathcal{V}_{i,:} = (\mathcal{V}_{i,1}, ..., \mathcal{V}_{i,K}) \in \mathbb{R}^K$ is used to denote the data located in the pixel with coordinates $i \in \mathbb{Z}^2$. Here, an eight-connected neighborhood is used and represented by the finite difference system

$$\mathcal{N} = \{(1,0), (0,1), (1,1), (1,-1)\} \tag{5.6}$$

with the weights $\omega_{1/2} = \sqrt{2} - 1$ and $\omega_{3/4} = 1 - \frac{\sqrt{2}}{2}$. For details, see [32, 158].

5.3.2 Relaxation

Since solving Eq. (5.3) is computationally extremely hard, the following simplifications are imposed to make it tractable: For the feature learning part, it is proposed to replace the strict ℓ_0-term in Eq. (5.5) by the smooth non-convex sparsity promoting surrogate function

$$\|\nabla_a\mathcal{V}\|_{0,\nu} = \sum_i \log(1 + \nu\|\mathcal{V}_{i,:} - \mathcal{V}_{i+a,:}\|_2^2), \tag{5.7}$$

which is a good approximation of the jump penalty with equality in the limit of its parameter ν, as was shown in Chapter 3.2. Further, let $\varepsilon = 0$ in Eq. (5.3) which leads to minimizing

the (preliminary) cost function

$$f(\boldsymbol{\Phi}) = \sum_{s=1}^{S} \omega_s \|\nabla_{a_s} \sigma(\boldsymbol{\Phi}\boldsymbol{U})\|_{0,\nu} \tag{5.8}$$

for learning the filters. It is noted, that the latter assumption removes the necessity to perform segmentation during learning and thus allows to proceed sequentially instead of in an alternating way.

The non-linear transformation σ of the filter responses in Eq. (5.8) is realized via $\sigma(x) := \log(1 + \mu x^2)$ with parameter $\mu > 0$. Note, that σ is smooth and symmetric, and that it allows to attenuate outliers in the filter responses.

5.3.3 Learning from patch samples

The training samples are chosen as a subset of image locations (and not all patches given by the image). This can be motivated as follows: first, when learning convolutional filters by minimizing Eq. (5.8), the inner products of each filter kernel $\boldsymbol{\Phi}_k$ with the pixel neighborhood at all image locations (i, j) are evaluated and summed up w.r.t. i, j. Due to overlap, calculating the whole sum results in redundant computations. Secondly, since the data of interest consists of texture segments, repeating patterns are expected which make the full patch set even more redundant. Based on this intuition, a randomly sampled subset of patches should suffice to learn the features from a texture image. Hence, only a fixed number $M << h \cdot w$ of randomly sampled patches are considered as training set.

Formally, the data in the objective function in Eq. (5.8) is modified from the jump set over features of the entire image to the empirical mean of a set of randomly sampled super-patches $\boldsymbol{U}_i$ and as a result, one obtains

$$f(\boldsymbol{\Phi}) = \frac{1}{M} \sum_{i}^{M} \sum_{s=1}^{S} \omega_{a_s} \|\nabla_{a_s} \sigma(\boldsymbol{\Phi}\boldsymbol{U}_i)\|_{0,\nu}. \tag{5.9}$$

Here, the super-patches' support templates are extensions of the $\sqrt{n} \times \sqrt{n}$ support template of the filters which additionally take the considered finite difference stencil into account,

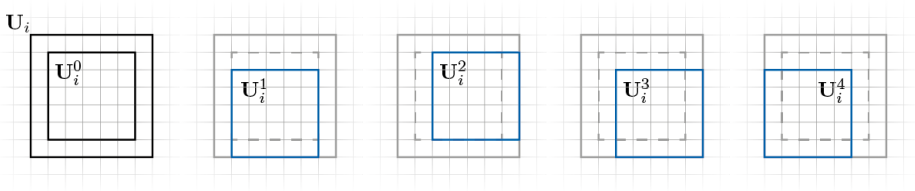


Figure 5.2: Illustration of an extracted super patch $\boldsymbol{U}_i$ and its neighboring patches $\boldsymbol{U}_i^{a_s}$ with respect to the utilized finite difference system $\mathscr{N}$.

as visualized in Figure 5.2. For first order finite differences, a corresponding one pixel neighborhood of the considered $\sqrt{n}\times\sqrt{n}$ template is sufficient. Different crops are generated from these super-patches according to the direction of the finite difference discretization and evaluate the inner products w.r.t. these crops (and apply σ). Finally, the respective finite difference operator is applied to the obtained result.

5.3.4 Constraints

In order to avoid trivial solutions such as the zero kernel and redundancies, several constraints are imposed on the filters.

Norm and coherence constraints

Analogous to Chapters 3 and 4, norm and coherence constraints are employed. To prevent the filter coefficients from shrinking to zero, the Euclidean norm of each filter is required to equal one, i.e.

$$\|\boldsymbol{\Phi}_k\|_2 = \sqrt{\sum_{i=1}^{n}(\boldsymbol{\Phi}_k)_i^2} = 1, \qquad k = 1,\dots,K. \tag{5.10}$$

Here, n is the number of coefficients in a single filter. For brevity, only 2D filters of quadratic support with size $\sqrt{n}\times\sqrt{n}$ are considered. The extension to filters supported on a rectangle is obvious. Geometrically, the norm constraint implies that each filter is an element of the $(n-1)$-dimensional sphere $\mathcal{S}_{n-1}$ in $\mathbb{R}^n$, and that the filter set constitutes a product of K such spheres. This structure is commonly referred to as oblique manifold, i.e. matrices in

$\mathbb{R}^{n\times K}$ with normalized columns, denoted by

$$\boldsymbol{\Phi}^\top \in \mathcal{S}_{n-1}^{\times K}. \tag{5.11}$$

In addition, the coherence penalty from Eq. (3.10) is used,

$$r(\boldsymbol{\Phi}) = -\sum_{1\le i\le j\le K} \log(1 - \langle \boldsymbol{\Phi}_i, \boldsymbol{\Phi}_j\rangle^2) \tag{5.12}$$

to well separate these vectors on the sphere. In particular, this soft constraint avoids pairwise collinear filters. It is pointed out that a minimum of that function is clearly achieved if the filters are orthogonal to each other, i.e. if the filter set lies in the corresponding Stiefel manifold. However, in the context of sparse coding, imposing such orthogonality directly as a hard constraint has turned out to be too restrictive (see Chapter 2).

Zero-mean constraint

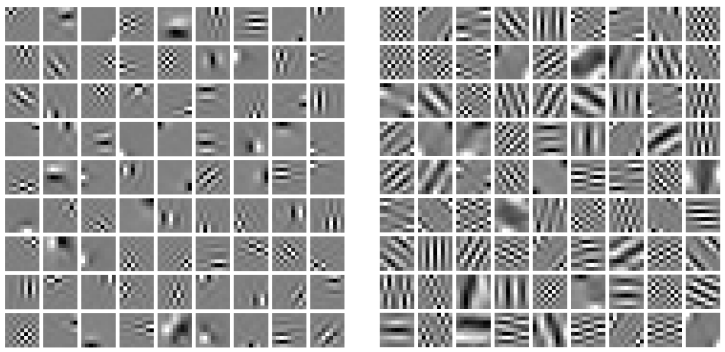
The mean over the patch is a distinguished feature with special discriminative power. It is considered as a seeded filter in the filter bank and the other filters are learned in its orthogonal complement. This means that filters with vanishing first order moments are learned, i.e. filters whose coefficients sum up to zero,

$$\sum_{i=1}^{n} \boldsymbol{\Phi}_{k,i} = 0. \tag{5.13}$$

It should be noted that these filters are oblivious to the patch mean which might vary, for instance due to small differences in lighting or contrast. Geometrically, the filters that satisfy Eq. (5.13) are contained in the hyperplane which contains the origin and which is orthogonal to $\mathbb{1}_n = (1,\dots,1)$. Hence, the set of feasible solutions forms the same Riemannian manifold as in Eq. (3.8), recollecting its definition

$$\mathcal{R} = \left(\mathcal{S}_{n-1} \cap \mathbb{1}_n^\perp\right)^{\times K}. \tag{5.14}$$

The Riemannian structure is important for the optimization procedure used later on.



(a) **(b)**

Figure 5.3: Effect of the proposed central moment constraint. Two sets of filters learned from the same gray-scale cartoon image. (Dark and light pixels represent negative and positive filter coefficients respectively, while neutral grey indicates coefficients equal or close to zero.) The filters in *(a)* were learned with the coherence constraint from Eq. (5.12) but without the centroid constraint in Eq. (5.16). In contrast, the filters in *(b)* were learned using both the coherence constraint Eq. (5.12) and the central moment constraint Eq. (5.16). It is clearly visible that the effective support sizes of many filters in *(a)* are in fact much smaller than 9×9, and that some shifted versions of the same filter can be identified among all filters. These undesirable effects are significantly reduced in *(b)*.

Central moment constraint

It might happen that there are two minimizers of Eq. (5.8), which adhere to norm and coherence constraints, and which are shifted versions of each other, see Figure 5.3a. There, it can be seen that the effective support size of many filters is much smaller than the prescribed maximum 9×9 filter size.

To avoid learning shifted versions of the same filter, a constraint on the centroid of the squared filter coefficients is proposed. Intuitively, by penalizing off-centered centroids of the pointwise squared (real-valued) filters, learning of filters that are shifted versions of their centered twin is avoided. To be more precise, considering the filter $\boldsymbol{\Phi}_k$, it can be noticed that by the employed normalization one obtains $\hat{\boldsymbol{\Phi}}_k^\top \hat{\boldsymbol{\Phi}}_k = \sum_i \sum_j (\boldsymbol{\Phi}_k)_{ij}^2 = 1$ where $\hat{\boldsymbol{\Phi}}_k$ denotes the vectorized 2D filter $\boldsymbol{\Phi}_k$. Thus, the pointwise square $\boldsymbol{\Psi}_k$ defined by $(\boldsymbol{\Psi}_k)_{ij} = (\boldsymbol{\Phi}_k)_{ij}^2$ can be viewed as a discrete 2D probability distribution. Hence, the components of the center of mass of this distribution may be computed by

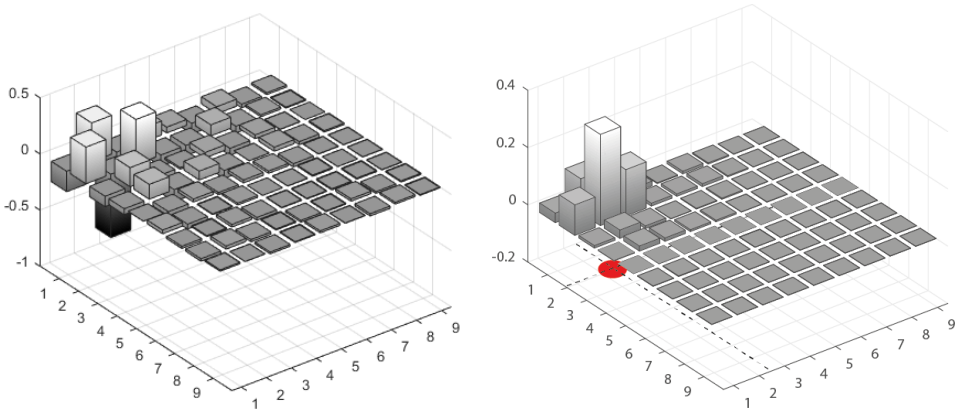
$$\bar{c}_{k,x} = \hat{\boldsymbol{\Phi}}_k^\top \boldsymbol{P}_x \hat{\boldsymbol{\Phi}}_k, \quad \bar{c}_{k,y} = \hat{\boldsymbol{\Phi}}_k^\top \boldsymbol{P}_y \hat{\boldsymbol{\Phi}}_k. \tag{5.15}$$

Here $\boldsymbol{P}_x$ is a diagonal matrix realizing the first moment with respect to the x-direction $\sum_{ij} i(\boldsymbol{\Psi}_k)_{ij}$, and $\boldsymbol{P}_y$ is given analogously.

Further, the normalization $c_{k,x} = (\bar{c}_{k,x} - \frac{\sqrt{n}+1}{2})/\frac{\sqrt{n}-1}{2})$ is employed and the analogous normalization for $c_{k,y}$ to obtain quantities $c_{k,x}, c_{k,y}$ centered at 0 with range between -1 and 1. For a filter centered around the origin, one requires $c_{k,x}, c_{k,y}$ to be close to zero. To this end, here the (convex) penalty

$$h(\boldsymbol{\Phi}) = \sum_{k=1}^{K} -\log[(1 - c_{k,x}^2)(1 - c_{k,y}^2)] + \frac{1}{2}(c_{k,x} - c_{k,y})^2. \tag{5.16}$$

is introduced. The effects of the central moment conditions are illustrated in Figure 5.3 and in Figure 5.4.



(a) **(b)**

Figure 5.4: The coefficients $(\boldsymbol{\Phi}_1)_{ij}$ of the first filter $\boldsymbol{\Phi}_1$ from the learned set $\boldsymbol{\Phi}$ depicted in Figure 5.3a without the centroid constraint (a) and its mass distribution (b). The red circle denotes the centroid $(c_{1,x}, c_{1,y})$.

5.3.5 Simplified learning problem and numerical optimization

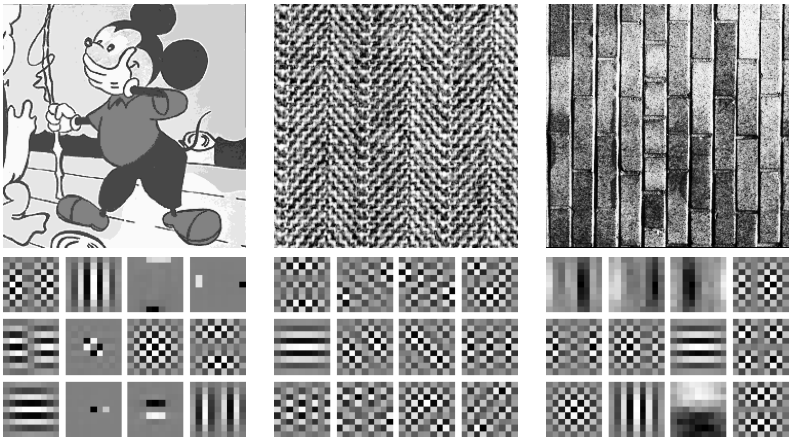
Summing up the considerations of this section, the relaxed objective in Eq. (5.9) is proposed with the soft coherence constraint from Eq. (5.12) and the soft shift constraint from Eq. (5.16) which reads as

$$E(\boldsymbol{\Phi}) = f(\boldsymbol{\Phi}) + \lambda r(\boldsymbol{\Phi}) + \kappa h(\boldsymbol{\Phi}). \tag{5.17}$$

Here, λ and κ are positive parameters. The soft coherence constraint r makes the filters disentangled. The soft shift constraint h pulls the center of mass of the filters towards the origin. The learning objective in Eq. (5.17) is a smooth non-convex function. The hard constraints (norm constraints, vanishing first moments) are encoded in the manifold $\mathcal{R}$ defined by Eq. (5.14). Equipped with this notation, the learning task reads as

$$\boldsymbol{\Phi}^\star \in \operatorname*{argmin}_{\boldsymbol{\Phi}^\top \in \mathcal{R}} E(\boldsymbol{\Phi}). \tag{5.18}$$

In order to solve Eq. (5.18) numerically, efficient schemes that exploit the geometric structure of the manifold $\mathcal{R}$ are required. The similarity of the geometric structure of the problem allows to use the same geometric optimization framework as in previous chapters, which is summarized in Appendix A. The solver requires the gradient of Eq. (5.18) with respect to the filters $\boldsymbol{\Phi}$. The derivation of this gradient is provided in Appendix B.2. The procedure is started with a random initialization in $\mathcal{R}$ and iterated until the Frobenius norm of the Riemannian gradient falls below the threshold of 10^{-5}. For illustration purposes, Figure 5.5 depicts the learned filter sets for different images.



(a) cartoon image **(b)** Brodatz 4 **(c)** Brodatz 12

Figure 5.5: Filter sets (bottom) learned from different textured images (top).

5.3.6 Extension to vector-valued images

So far, only gray-scale texture images were considered whereas textured images often have multiple channels, for instance RGB color images. The proposed method is subsequently extended for the case when the image $\boldsymbol{U}$ is vector-valued with L channels, that is, if $\boldsymbol{U}_{ij} \in \mathbb{R}^L$. Let $\boldsymbol{U}_{i,l}^{a_s}$ the i-th patch cropped according to direction a_s in channel $l \in 1, \ldots, L$. Intuitively, different channels of an image should require different filter sets such that spatial homogeneity of filter responses can be achieved. To that end, first the formulation of the patch-based filter operation is extended

$$\boldsymbol{\Phi}_k \boldsymbol{U}_i^{a_s} = \begin{bmatrix} \boldsymbol{\Phi}_{k,1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \boldsymbol{\Phi}_{k,L} \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_{i,1}^{a_s} \\ \vdots \\ \boldsymbol{U}_{i,L}^{a_s} \end{bmatrix}. \tag{5.19}$$

In this work, RGB images are considered as examples of multi-channel images. Since the red, the green and the blue channels are in general highly correlated, it can be safely assumed that the patch structure within each channel will be similar and set $\boldsymbol{\Phi}_{k,R} = \boldsymbol{\Phi}_{k,G} = \boldsymbol{\Phi}_{k,B} = \boldsymbol{\Phi}_k$. Thus, the learned filters act on the different channels in the same way. It is noted that this does not hinder jumps in a single channel to be detected.

5.4 Segmentation stage

Having explained the relaxation of the model from Eq. (5.3) to determine suitable filters $\boldsymbol{\Phi}$, the segmentation given a set of filters is discussed next. To segment the vector-valued feature image $\sigma(\mathcal{F})$, consider the (formal) Lagrangian version of the discretization of Eq. (5.3) for fixed $\boldsymbol{\Phi}$ to obtain the problem

$$\underset{\mathcal{V}}{\operatorname{argmin}} \ \gamma \sum_{s=1}^{S} \omega_s \|\nabla_{a_s} \mathcal{V}\|_0 + d(\mathcal{V}, \sigma(\mathcal{F})). \tag{5.20}$$

Here, $\gamma > 0$ is a parameter for tuning the trade-off between data fitting and regularity, and $\mathcal{F} = \boldsymbol{\Phi}\boldsymbol{U}$ are again the filter responses with respect to the image $\boldsymbol{U}$.

5.4.1 Filter weighting based on the Mahalanobis distance

Recall that all filters were constrained to have unit norm in the learning stage. As a result, all filter outputs are weighted equally regardless of their discriminative power. To account for this, a data fidelity term based on the Mahalanobis distance d is utilized. The covariance matrix of all feature vectors (after applying the non-linearity) is used. With slight abuse of notation, let

$$\boldsymbol{\Sigma} = \operatorname{cov}(\mathcal{G}) \tag{5.21}$$

represent the $K \times K$ covariance matrix of all feature vectors in $\mathcal{G} = \sigma(\mathcal{F})$. To define the corresponding Mahalanobis distance, one writes $\boldsymbol{\Sigma}\mathcal{G}$ for the action of a $K \times K$ matrix $\boldsymbol{\Sigma}$ on the third index of the feature image $\mathcal{G}$, i.e. $(\boldsymbol{\Sigma}\mathcal{G})_{ijk} = (\boldsymbol{\Sigma}\mathcal{G}_{ij})_k$. Then, the Mahalanobis data fidelity reads as

$$\begin{aligned} d(\mathcal{V}, \mathcal{G}) &= \sum_{ij} \|\boldsymbol{\Sigma}^{-1/2}(\mathcal{V}_{ij} - \mathcal{G}_{ij})\|_2^2 \\ &= \|\boldsymbol{\Sigma}^{-1/2}(\mathcal{V} - \mathcal{G})\|_2^2. \end{aligned} \tag{5.22}$$

It was observed that the results slightly improve when normalizing $\boldsymbol{\Sigma}^{-1/2}$ by $\max_{ij} (\boldsymbol{\Sigma}^{-1/2})_{ij}$.

5.4.2 Variational partitioning of the feature images

By the previous considerations in Section 5.4.1, the minimization problem from Eq. (5.20) with the Mahalanobis data term needs to be solved. Plugging $\mathcal{V} = \boldsymbol{\Sigma}^{1/2}\mathcal{U}$ into Eq. (5.20) yields the problem

$$\operatorname*{argmin}_{\mathcal{U}} \; \gamma \sum_{s=1}^{S} \omega_s \|\nabla_{a_s} \boldsymbol{\Sigma}^{1/2}\mathcal{U}\|_0 + \|\mathcal{U} - \boldsymbol{\Sigma}^{-1/2}\mathcal{G}\|_2^2. \tag{5.23}$$

An important observation is that the ℓ_0-prior is invariant to invertible matrices acting in the third dimension, i.e. $\|\nabla_{a_s} \boldsymbol{\Sigma}^{1/2}\mathcal{U}\|_0 = \|\nabla_{a_s}\mathcal{U}\|_0$. Therefore, the problem from Eq. (5.23) is

equivalent to

$$\mathcal{U}^* = \underset{\mathcal{U}}{\operatorname{argmin}}\ \gamma \sum_{s=1}^{S} \omega_s \|\nabla_{a_s} \mathcal{U}\|_0 + \|\mathcal{U} - \boldsymbol{\Sigma}^{-1/2}\mathcal{G}\|_2^2. \tag{5.24}$$

As it turns out, this constitutes a classical (vector-valued) piecewise constant Mumford-Shah problem for data $\boldsymbol{\Sigma}^{-1/2}\mathcal{G}$ with an ℓ_2-norm data term. This is a challenging optimization problem in its own, but there are well-working approximate strategies available. Here, the ADMM-based method developed in [159, 158] is used. Although computationally more demanding than other recent approaches [182, 38, 126], this method currently gives the best quality in practice, as was shown in the comparison [126].

5.4.3 Obtaining the label map

The result obtained from treating the problem in Eq. (5.22) is a vector-valued piecewise constant function. To obtain the final label map (scalar field), the vector sum in a pixel is simply used as (real-valued) index for a segment, i.e. the coefficients along the feature vector are summed up at every pixel location. It can be observed that segment boundaries often lead to high filter responses which result in small spurious segments at the boundaries. To remove these, the simple post-processing step from [188] is used, where small regions are merged with neighbors based on their boundary ratios. It must be noted that this merging

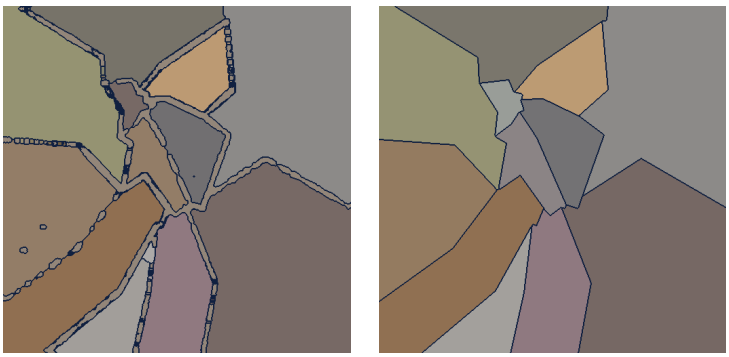


Figure 5.6: In a postprocessing step, small spurious segments are merged into their neighboring segments. *Left:* Raw segmentation. *Right:* final segmentation after region merging.

is not a hierarchical approach. Figure 5.6 depicts the final segmentation before and after the boundary refinement step.

5.5 Experimental results

The proposed learning and segmentation method is implemented in Matlab. For the segmentation step described in Section 5.4.2, the toolbox Pottslab[1] was used. In addition, the region merging implementation from [188] was applied as post-processing. The experiments were conducted on a desktop computer with an Intel i7-3930K processor with 3.2 GHz.

The segmentation results produced by the proposed method are compared with existing algorithms on two different datasets. For a quantitative comparison, the well-known Prague texture segmentation dataset is used which comprises mosaics of color and grayscale textures. In addition, the same method is shown qualitatively to be also effective in segmenting the histology images from [115].

5.5.1 Prague texture dataset

The Prague texture segmentation dataset [69] consists of 80 texture mosaics which are synthetically generated from random compositions of 114 different textures from 10 thematic categories. Color (RGB) and grayscale versions of this dataset are available along with the respective ground truth segment map and each texture mosaic is of size 512×512 pixels and the number of segments varies between 3 and 12. For a quantitative comparison, segmentations of the large color texture dataset – used in the ICPR 2014 contest – are produced and evaluated against their ground truth using *(i)* region-based metrics: Correct Segmentation (CS), Over-Segmentation (OS), Under-Segmentation (US), Missed Error (ME), Noise Error (NE); *(ii)* pixel-based metrics: Omission error (O), Commission error (C), Class Accuracy (CA), recall (CO), precision (CC), type I error (I.), type II error (II.), mean class Accuracy Estimate (EA), Mapping Score (MS), Root Mean square proportion estimation error (RM), Comparison Index (CI); *(iii)* consistency-based metrics: Global Consistency Error (GCE)

[1] Available at `http://pottslab.de`.

and Local Consistency Error (LCE). If available, the Mirkin metric (dM), Van Dongen metric (dD) as well as the Variation of Information (dVI) are reported. For computing these metrics, the benchmark provided by the authors of the Prague dataset is used. Please see [70], where a detailed definition of above metrics can be found.

For each of the 80 texture mosaics in the benchmark, a separate set of filters $\boldsymbol{\Phi}$ is learned and the segmentation based on these filter outputs is computed subsequently. The parameters for learning the features and performing the Potts segmentation are set empirically and remain fixed for all instances in the dataset. The learned filter sets contain $K = 41$ filters of size 9×9 each and are learned from $M = 50\,000$ patches that are drawn from the mosaic (uniform random sampling). By setting the parameter M to a large value, effectively all patches of the image are considered for learning. However, it was observed that results did not improve beyond $M = 50K$. In principle, the objective function in Eq. (5.3) does not require the filters $\boldsymbol{\Phi}_1, \ldots, \boldsymbol{\Phi}_K$ to be of equal size. For simplicity, filters of identical size were used, and it is noted that filters of smaller size are included in the utilized filter set by zero-padding. As is common practice in patch-based methods (for example [127]), all pixels in the patch are weighted by a Gaussian mask to give more weight to the central pixel which leads to slightly better localized segment boundaries. In the learning problem of Eq. (5.18) the parameters of the non-linearities are set to $\mu = \nu = 2000$ and the weights of the coherence and moment-centering penalties to $\lambda = 10$ and $\kappa = 10$. In the Potts segmentation that follows, a weight is required that trades data fidelity against spatial homogeneity of the solution and therefore effectively influences the degree of over-segmentation. Empirically, $\gamma = 0.03$ is found to provide a good trade-off between over- and under-segmentation over all benchmark images. The texture mosaic needed in average 35 min for the learning stage and 9 min for the segmentation stage.

To assess the performance of the proposed approach, the obtained results are compared to several state-of-the-art algorithms that were used for segmentation on the Prague texture mosaics such as the Texel-based Segmentation (TS) [164], Segmentation by Weighted Aggregation (SWA) [61], Gaussian MRF Model (GMRF) with Expactation Maximization (EM) [71], 3-D Auto Regressive Model with EM (AR3D) [68], Texture Fragmentation and Reconstruction (TFR) and (TFR+) [153], Regression-based Segmentation (RS) [189], Factorization-Based Texture Segmentation (FSEG) [188], Priority Multi-Class Flooding Algorithm (PM-

Method	TS	SWA	GMRF	AR3D	TFR	TFR+	RS	FSEG	PMCFA	PCA-MS	Proposed
↑ CS	59.13	27.06	31.93	37.24	46.13	51.25	46.02	69.02	75.32*	72.27	**77.73**
↓ OS	10.89	50.21	53.27	59.53	**2.37**	5.84*	13.96	17.30	11.95	18.33	15.92
↓ US	18.79	**4.53**	11.24	8.86	23.99	7.16	30.01	11.85	9.65	9.41	6.31*
↓ ME	10.45	25.76	14.97	12.54	26.70	31.64	12.01	6.28	4.57	4.19*	**3.93**
↓ NE	9.93	27.50	16.91	13.14	25.23	31.38	11.77	5.66	4.63	**3.92**	**3.92**
↓ O		33.01	36.49	35.19	27.00	23.60	35.11	10.79	**4.51**	7.25*	7.68
↓ C		85.19	12.18	11.85	26.47	22.42	29.91	13.75	8.87*	**6.44**	24.24
↑ CA		54.84	57.91	59.46	61.32	67.45	58.75	77.50	**83.50**	81.13	82.80*
↑ CO		60.67	63.51	64.81	73.00	76.40	68.89	84.11	**88.16**	85.96	86.89*
↑ CC		88.17	89.26	91.79*	68.91	81.12	69.30	86.89	90.73	91.24	**93.65**
↓ I.		39.33	36.49	35.19	27.00	23.60	31.11	15.89	**11.84**	14.04	13.11*
↓ II.		2.11	3.14	3.39	8.56	4.09	8.63	2.60	**1.47**	1.59	1.50*
↑ EA		66.94	68.41	69.60	68.62	75.80	65.87	83.99	**88.10**	87.08	88.03*
↑ MS		53.71	57.42	58.89	59.76	65.19	55.52	78.25	**83.98**	81.84	**83.98**
↓ RM		6.11	4.56	4.66	7.57	6.87	10.96	4.51	3.76*	4.45	**3.27**
↑ CI		70.32	71.80	73.15	69.73	77.21	67.35	84.71	88.74*	87.81	**89.03**
↓ GCE		17.27	16.03	12.13	15.52	20.35	11.23	10.82	**6.51**	8.33	7.40*
↓ LCE		11.49	7.31	6.69	12.03	14.36	7.70	7.51	**3.92**	5.61*	5.62
↓ dD							18.52		10.13	9.06*	**8.57**
↓ dM							23.67		6.41	5.88*	**5.30**
↓ dVI							**13.31**		15.80	14.54*	14.88

Table 5.1: Results on the Prague Color Texture Dataset (ICPR2014 Contest). Each row corresponds to a segmentation quality metric, and the arrow indicates if high ↑ or low values ↓ are better. The first rank is marked by boldface, the second rank is marked by an asterisk.

CFA) [136, 137] and Variational Multi-Phase Segmentation (PCA-MS) [117]. Table 5.1 provides the segmentation accuracy benchmark results as reported on the benchmark website [70] and in [188] as well as in [117]. In addition, Figure 5.7 depicts some of the segmentations produced by the four top-performing methods including the results of the proposed method for visual comparison.

5.5.2 Parameter sensitivity

In the following, the sensitivity of the proposed method with respect to the most influential parameters is explored. To that end, an evaluation of the method is conducted with varying parameters on a representative subset of images from the Prague benchmark dataset drawn from the different categories all, bark, flowers, glass, nature, stone and textile. The parameters filter size, the number of learned filters K, the weight of the filter coherence penalty λ,

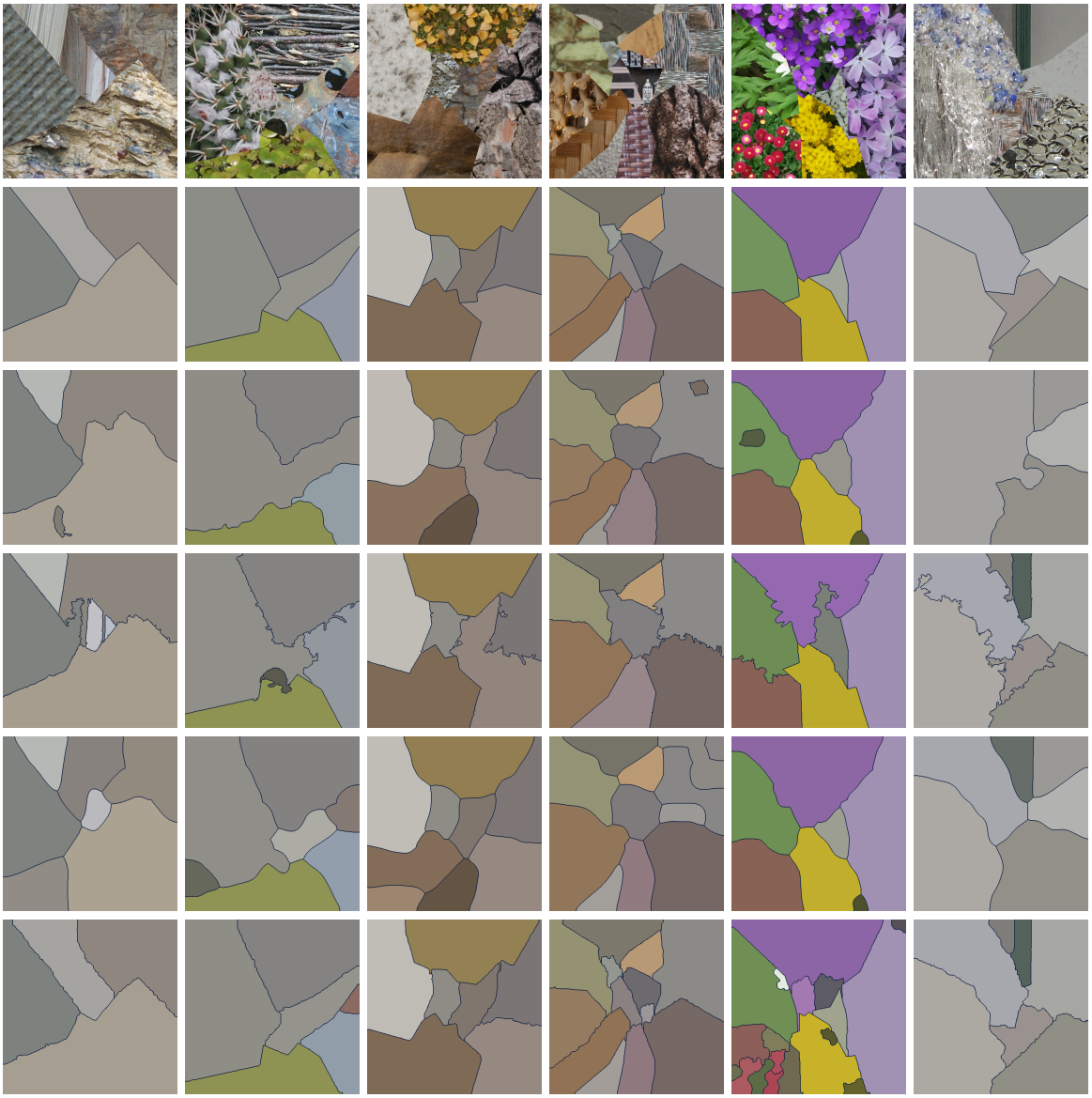


Figure 5.7: Exemplary segmentation results on the Prague texture segmentation dataset. *From top to bottom:* input image, ground truth, FSEG [188], PMCFA [136, 137], PCA-MS [117] and the proposed method.

and the parameters μ and ν of the employed non-linearities are examined. Each of them is varied while keeping the others fixed at the values described in Subsection 5.5.1.

The evaluation begins with the parameter λ which controls the maximum coherence between all filter pairs and which is given in Eq. (5.18). Table 5.2 shows that if λ is close to zero which effectively disables this constraint, segmentation results deteriorate significantly. For λ larger than 1, only negligible changes in segmentation results across all quality metrics are observed. These results underline the importance of the constraint in the proposed learning objective but also reveal that the choice of its exact value is not critical as long as it is large enough.

λ	0.1	1	5	10	15	20	50	100
↑ CS	57.38	72.58	70.97	71.59	72.54	65.85	68.56	75.70
↓ OS	9.67	18.64	18.69	18.73	18.63	18.71	18.67	14.81
↓ US	34.65	7.98	3.66	7.78	3.67	14.58	11.72	7.98
↓ ME	2.97	5.19	10.04	9.26	10.04	5.19	5.19	5.19
↓ NE	0.42	5.97	10.42	10.13	10.41	6.14	6.23	5.96
↓ O	26.14	9.44	12.12	12.97	10.04	11.57	11.36	9.34
↓ C	26.22	31.05	31.07	31.19	31.03	31.84	33.88	31.12
↑ CA	68.49	79.52	78.90	79.59	79.75	75.48	77.23	81.27
↑ CO	78.31	83.99	83.13	84.34	84.14	81.47	82.55	85.80
↑ CC	74.22	90.96	91.19	90.45	90.77	86.06	88.20	90.96
↓ I.	21.69	16.01	16.87	15.66	15.86	18.53	17.45	14.20
↓ II.	5.00	1.49	1.38	1.54	1.52	1.99	1.88	1.52
↑ EA	74.10	85.14	84.67	85.41	85.37	81.54	83.22	86.58
↑ MS	69.46	79.69	79.14	79.79	79.78	75.78	77.27	81.47
↓ RM	7.32	3.85	3.83	3.79	3.73	4.71	4.09	3.59
↑ CI	75.11	86.22	85.81	86.33	86.33	82.58	84.23	87.42
↓ GCE	6.34	8.87	8.69	9.86	8.80	9.58	9.78	8.77
↓ LCE	5.31	5.90	5.99	6.83	6.12	5.95	6.05	5.78
↓ dD	12.76	9.99	10.40	10.21	9.93	11.19	10.71	9.06
↓ dM	11.19	5.39	5.48	5.47	5.31	6.10	5.87	4.96
↓ dVI	14.02	15.37	15.50	15.27	15.33	15.15	15.28	15.17

Table 5.2: Sensitivity w.r.t. the filter coherence penalty λ

Next, the influence of the number of filters K on the results is investigated. From Table 5.3 it can be concluded that the segmentation quality increases for up to 41 filters and deteriorates for larger numbers. The initial improvement might be explained by the increased discriminatory power obtained from a larger number of different filters. The deterioration of

K	11	21	31	41	61	81	122	162
↑ CS	32.37	52.23	70.25	72.59	59.13	43.36	17.43	0.47
↓ OS	6.93	13.94	15.13	18.73	14.74	16.10	21.06	0.00
↓ US	66.81	28.78	13.65	7.78	8.49	3.65	30.43	46.97
↓ ME	0.15	14.43	11.91	5.19	16.14	35.38	29.49	51.15
↓ NE	0.00	14.29	12.53	6.18	15.99	35.68	27.38	47.06
↓ O	51.79	42.58	14.76	10.17	17.22	27.10	52.82	76.49
↓ C	36.68	31.94	22.52	31.26	31.78	46.46	76.02	64.75
↑ CA	44.39	61.58	75.33	80.02	71.65	63.04	38.22	21.89
↑ CO	59.30	71.38	82.67	84.27	77.58	70.48	49.76	38.53
↑ CC	46.42	68.23	81.84	91.25	88.68	84.58	68.67	42.50
↓ I.	40.70	28.62	17.33	15.73	22.42	29.52	50.24	61.47
↓ II.	14.82	6.25	2.83	1.38	2.16	3.69	9.53	15.08
↑ EA	49.77	67.15	80.61	85.61	78.86	71.84	49.06	31.98
↑ MS	39.39	58.60	75.53	80.28	72.36	62.84	32.61	11.44
↓ RM	17.04	10.77	5.81	3.82	5.27	6.13	14.13	18.71
↑ CI	51.20	68.39	81.38	86.62	80.79	74.35	53.51	35.65
↓ GCE	3.21	7.71	7.75	8.65	12.27	17.57	19.34	25.64
↓ LCE	2.27	5.35	6.46	5.88	7.88	12.06	14.01	17.93
↓ dD	21.09	16.48	11.23	9.80	14.39	19.70	30.97	37.73
↓ dM	38.36	18.27	7.81	5.28	7.77	11.91	31.97	43.74
↓ dVI	11.81	13.52	14.41	15.34	15.85	16.05	15.42	13.46

Table 5.3: Sensitivity w.r.t. the number of filters K

the quality for a larger number of filters might be explained by an over-segmentation caused by irrelevant features.

The study is continued with the influence of the filter size. The choice of the filter size should relate to the scale of the texture. Although the Prague texture mosaics expose a relatively large variety of texture scales, filter sizes of 7 and 9 pixels were found to achieve the best results in average (see Table 5.4), which confirms the choice of other works, e.g. [117, 188]. For small filter sizes, within-texture variations are similar to variations at texture boundaries which leads to under-segmentation in the segmentation stage. For the proposed method, it can also be observed that large filters lead to a decreased localization of texture boundaries, and to larger spurious segments at texture boundaries as depicted in Figure 5.6.

Next, the non-linearity parameter ν is considered which is used for relaxation of the ℓ_0-jump penalty for filter learning in Eq. (5.7). Recalling that for large ν the surrogate function approximates the original sparsifying function well [5, 127]. Table 5.5 lists segmentation results over a large range of ν. The segmentation fails for small values of ν and improves

filter size	5	7	9	11	13
↑ CS	55.23	69.74	71.59	59.68	54.99
↓ OS	9.22	14.05	18.73	13.06	8.56
↓ US	39.69	14.28	7.78	16.11	9.31
↓ ME	6.63	8.39	9.26	17.14	25.10
↓ NE	5.93	7.22	10.13	17.95	25.22
↓ O	38.39	6.18	12.97	12.81	21.91
↓ C	21.64	16.97	31.19	33.25	32.46
↑ CA	62.27	78.59	79.59	71.45	66.58
↑ CO	73.21	84.33	84.34	78.28	74.54
↑ CC	64.40	87.46	90.45	86.10	81.01
↓ I.	26.79	15.67	15.66	21.72	25.46
↓ II.	8.39	1.66	1.54	2.30	3.55
↑ EA	66.87	83.50	85.41	78.82	74.44
↑ MS	60.30	78.59	79.79	72.27	66.01
↓ RM	10.50	5.42	3.79	5.64	6.31
↑ CI	67.78	84.57	86.33	80.37	76.00
↓ GCE	4.07	7.60	9.86	12.27	15.64
↓ LCE	3.59	4.80	6.83	8.01	11.30
↓ dD	14.56	9.97	10.21	14.00	17.09
↓ dM	21.38	5.96	5.47	7.75	10.35
↓ dVI	12.90	14.79	15.27	15.58	15.51

Table 5.4: Sensitivity w.r.t. the filter size

when increasing it. Due to the decreasing slope of the surrogate function for large values of ν, the learning algorithm converges more slowly. The choice in these experiments reflects a trade-off between convergence speed and approximation accuracy.

Finally, the sensitivity of the corresponding parameter μ in the non-linearity σ of Eq. (5.8) is investigated in the results of Table 5.6. The overall segmentation results are found to be robust over a large range of choices of μ and segmentation quality only starts suffering for very large values of μ where the shape of σ degenerates.

5.5.3 Histology dataset

In addition to the texture segmentation benchmark, the proposed method is applied to the histology dataset used in [115]. The dataset contains 36 color images of size 128×128 pixels of stained tissue along with segmentations by an expert. Instead of the adaptive color quantization used in [115], the images are simply converted to gray-scale prior to processing

ν	100	500	1000	2000	3000
↑ CS	0.00	29.57	49.31	71.56	69.02
↓ OS	0.00	0.00	6.83	18.71	23.61
↓ US	99.93	69.31	47.39	7.69	7.99
↓ ME	0.00	0.00	0.00	10.04	3.10
↓ NE	0.00	0.00	0.00	10.42	3.86
↓ O	100.00	58.36	44.93	10.50	13.68
↓ C	71.63	40.48	46.06	31.09	32.54
↑ CA	8.43	38.42	53.34	78.44	77.09
↑ CO	28.05	54.02	64.42	83.80	81.35
↑ CC	8.44	39.29	57.00	87.97	91.25
↓ I.	71.95	45.98	35.58	16.20	18.65
↓ II.	24.62	16.39	13.10	2.65	1.40
↑ EA	12.80	43.06	57.62	84.16	83.57
↑ MS	−7.92	31.03	47.78	78.78	77.82
↓ RM	30.45	19.14	15.15	4.40	4.36
↑ CI	15.25	44.71	59.04	84.97	84.86
↓ GCE	0.14	2.16	3.13	8.58	9.00
↓ LCE	0.14	1.78	2.55	6.04	6.20
↓ dD	36.01	23.55	18.61	10.04	11.23
↓ dM	75.36	43.91	35.86	7.01	5.93
↓ dVI	9.14	11.20	12.43	15.02	15.74

Table 5.5: Sensitivity w.r.t. the non-linearity parameter ν

for this experiment. It must be pointed out that this renders the problem more challenging due to the loss of color information. Since the images are considerably smaller than the Prague texture mosaics, the size of the learned filters is reduced to 5×5, only 13 filters are learned and the trade-off parameter in the segmentation stage is adjusted to a fixed $\gamma = 0.8$. Otherwise, the same setup as in the Prague texture experiment is used. It has to be emphasized that switching to this quite different class of images only required the adjustment of these few parameters. The learning stage requires approximately 2 minutes and the segmentation stage around 3 seconds. Some of the results are given in Figure 5.8.

5.5.4 Discussion

From Table 5.1 it can be observed that the proposed method significantly improves upon most existing approaches in the Prague texture segmentation benchmark. Moreover, the segmentations obtained by the proposed method are competitive with the previously best performing method PMCFA. PMCFA and the proposed method yield a comparable num-

μ	10	100	2000	5000	10000
↑ CS	74.69	71.79	72.59	65.99	24.30
↓ OS	18.44	22.18	18.73	15.33	6.80
↓ US	7.97	7.89	7.78	6.89	36.01
↓ ME	0.00	0.00	5.19	16.82	29.83
↓ NE	0.66	0.75	6.18	16.76	29.64
↓ O	13.33	13.96	10.17	15.18	59.31
↓ C	31.08	31.16	31.26	30.92	51.44
↑ CA	79.50	79.04	80.02	72.85	41.24
↑ CO	83.65	82.92	84.27	79.35	53.97
↑ CC	91.40	91.46	91.25	84.99	60.72
↓ I.	16.35	17.08	15.73	20.65	46.03
↓ II.	1.43	1.39	1.38	2.73	10.29
↑ EA	84.96	84.83	85.61	78.93	49.49
↑ MS	80.21	79.55	80.28	72.51	36.61
↓ RM	2.95	3.79	3.82	5.22	14.07
↑ CI	86.12	85.93	86.62	80.42	52.84
↓ GCE	8.22	8.12	8.65	13.34	15.58
↓ LCE	5.60	5.83	5.88	8.56	12.02
↓ dD	9.85	10.19	9.80	13.50	27.86
↓ dM	5.31	5.41	5.28	8.21	29.43
↓ dVI	15.59	15.57	15.34	15.26	14.53

Table 5.6: Sensitivity w.r.t. non-linearity parameter μ

ber of first and second ranks. The segmentation examples in Figure 5.7 indicate that the proposed method gives very satisfactory results for segments with clear repeated texture patterns, as for instance in the first three examples. Erroneous segmentations appear mostly when quite different patterns such as the red blossoms on green background in the fifth image are present in a segment. A possible cause for this is that the blossoms are interpreted as a texture on its own on a smaller scale. Qualitatively, one observed that the proposed method tends to a slight over-segmentation when large color contrasts are present. This is not the case for PMCFA. Compared to PMCFA, the proposed method produces smoother boundaries.

In addition to the Prague texture segmentation benchmark, the algorithm produces useful segmentations of the tissue images of [115]. It is mostly very close to the expert annotations. The fact that it was only necessary to adapt the maximum number of filters K, their maximum size and the segmentation hyperparameter γ to obtain the presented results, underlines the usefulness of this learning based method and validates the original idea. It also indicates the potential of the proposed method for segmenting different classes of texture images.

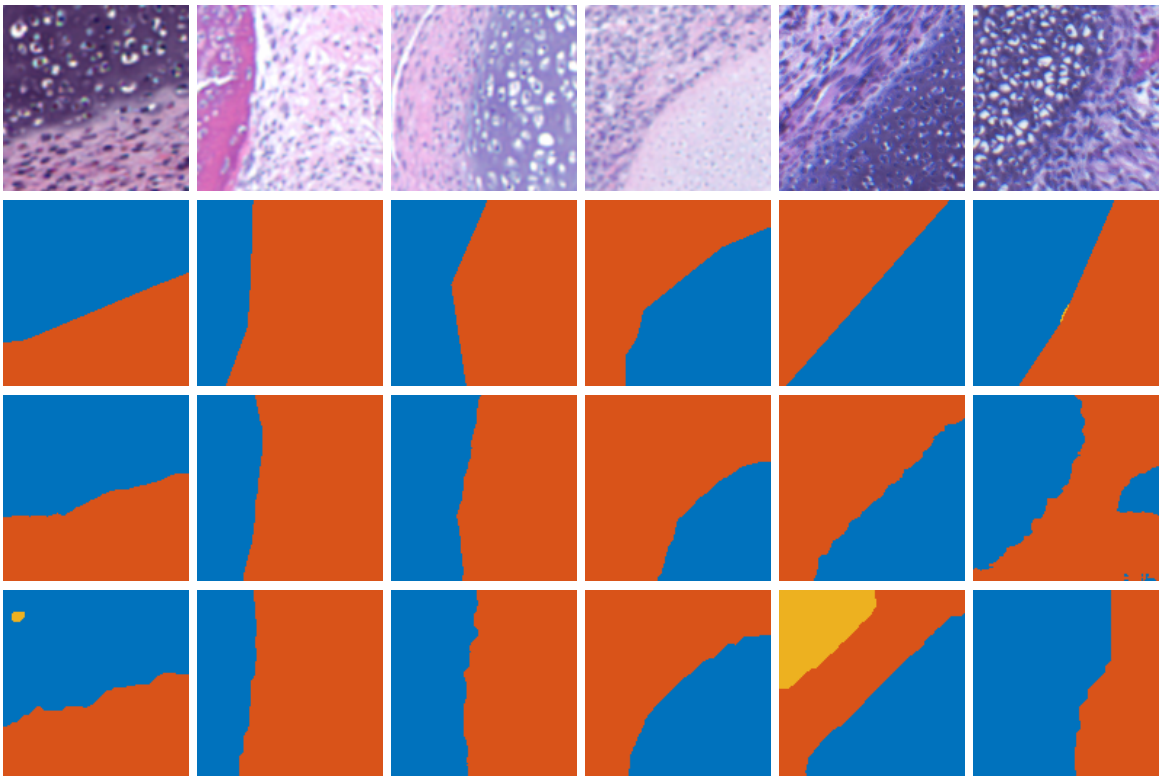


Figure 5.8: Segmentation results on the histology dataset from [115]. *From top to bottom:* input image, ground truth, ORTSEG [115], and the proposed method.

The main trade-off of the proposed method is currently a relatively long processing time per image. In contrast to most other methods where a fixed set of features is used for segmentation, executing the learning stage prior to segmentation is needed here, which increases the overall running time. The computational complexity of the learning stage is primarily determined by size and number of the filters as well as the number of patches and their channels that are used for learning. A speed-up could be achieved by reducing the number of training samples. It was observed that reducing the number of samples for training from $50K$ to $10K$ only slightly decreased the segmentation quality. Also, filter learning is so far started with a random initialization for every image. In practical applications, the filter set can be initialized with prelearned filters which could bring down the required number of iterations during learning and therefore significantly decrease the overall running time. A further speed-up might be obtained by an optimized implementation.

To summarize the results of this chapter, a method for unsupervised texture segmentation where the features are learned from images without ground truth segmentation was developed. The first main contribution is the design of a model for local image features based on the co-sparse analysis framework, which is extended with local homogeneity assumptions. The convolutional features are learned in a way that they produce approximately piecewise constant feature images and are combined with the piecewise constant Mumford-Shah model. The second main contribution is the development of a practical algorithm for unsupervised texture segmentation based on that model. To make the problem computationally tractable, it was relaxed and decomposed into a filter learning stage and a segmentation stage. In the filter learning stage, the geometric conjugate gradient descent method known from previous chapters was reused with respective adjustments. In the segmentation stage, on the other hand, the Lagrange formulation of the piecewise constant Mumford-Shah model was augmented with a Mahalanobis distance as data term. The proposed algorithm yields competitive results on the standard benchmark dataset for unsupervised texture segmentation. Furthermore, switching to the quite different class of histological images only required the adjustment of a few parameters. The improved segmentation quality underpins the idea of learning features adapted to the image under consideration. The proposed approach may be especially valuable in situations where creating large training sets of accurate ground truth segmentations or hand-crafting features is expensive.

Topics of future research include speeding-up the proposed method as explained in the discussion section as well as approaching the proposed non-smooth model more directly, that is, employing fewer relaxation steps.

Chapter 6

Software implementation

In previous chapters, the explanations of the developed approaches have focused to a large extent on the respective conceptual details as well as mathematical and numerical challenges, while software implementation details have only been touched upon in utter brevity. To achieve the presented results and apply the methods to real data, however, significant engineering effort is required. In this penultimate chapter, the concept of the software framework is presented, which was developed in the course of this thesis and which transfers the proposed models into operational computer programs.

The design of the software that implements the presented experiments is driven by the following principles. *(i)* All proposed methods are formalized data-related questions, formulated as mathematical problems, which are subsequently solved numerically. To enable fast prototyping and drive short iterations of experimentation, *extensibility* and *reusability* of the software components are central. The fewer changes have to be made in order to support a new cost function or setup a new experiment, the shorter the time required for debugging and the bigger the opportunity to optimize reusable parts. *(ii)* When dealing with numerical algorithms applied to large datasets, *computational efficiency* is an important aspect for implementation. In practice, a difference in milliseconds of execution time of an algorithm iteration can be decisive whether or not a suitable parameterization can be found and an idea works. Clearly, by keeping an implementation flexible for extensions, optimization of its computational efficiency becomes more challenging. The goal is to find a good trade-off between highly optimized but problem-specific and better generalizable but less efficient implementations. *(iii)* Collecting, logging and analyzing measurements from

all parts of the processing pipeline is paramount to translating research ideas into code effectively. A good software framework provides convenient interfaces for attaching sensors, visualizing and analyzing measurements and compiling data reports. In addition to these guiding principles, some other aspects are important. Portability of the implementation to different platforms is desirable to enable reproducibility of the achieved results for other researchers in the community. Configurability avoids change of code to parameterize the processing and algorithms for different experiments, which ensures consistency across experiment runs. Finally, support for distribution of the software package to multiple computing resources and automated collection of sub-results makes it possible to run more experiments in parallel and explore larger hyper-parameter spaces.

6.1 System overview

The technical realization of the presented methods involves several abstraction layers to implement the necessary computations on hardware. They can be roughly grouped into the three main layers: Operating System, Compute Engine and Application. *(i)* The *Operating System* (OS) layer implements low-level methods to manage and access hardware resources such as the processing units (CPU/GPU), the file system, main memory and network interfaces. It typically makes use of high-performance computing libraries that are tailored to the hardware underneath. Important for the presented algorithms are in particular linear algebra computing libraries, such as OpenBLAS [1], ATLAS [2], Math Kernel Library (MKL) [3] or CUDA [4]. The OS-layer abstracts the low-level algorithms and provides interfaces to upstream software components. *(ii)* The *Compute Engine* is responsible to translate mathematical operations in algorithms defined by the Application layer into efficient routines that process them numerically. Typically, libraries of the compute engine offer efficient implementations of common algorithms and interface with the low-level system libraries in the OS-layer. *(iii)* The top most abstraction layer is the *Application* layer which implements larger programs that orchestrate mathematical operations and data manipulations to com-

[1] `https://www.openblas.net`
[2] `http://math-atlas.sourceforge.net`
[3] `https://software.intel.com/mkl`
[4] `https://developer.nvidia.com/cuda-zone`

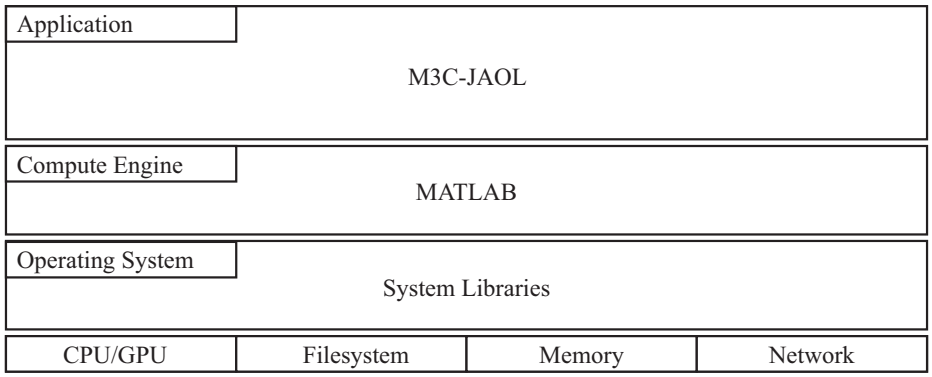


Figure 6.1: Overview of the system and its abstraction levels that compose the software implementation.

prise a full-fledged software package for experiments. Figure 6.1 sketches the overall system layout with its different layers of software abstraction.

To address the implementation considerations discussed in the previous section, the following design choices were made. For the lower layers, existing and proven implementations were chosen. Notably, all experiments were conducted on 64bit Windows or Linux operating systems equipped with MKL and CUDA libraries for efficient linear algebra operations on CPU and GPU. MATLAB by MathWorks Inc. was chosen as the computing engine, due to its extensive implementation library of signal and image processing algorithms, cross-platform support and good adoption in the research community. The entirety of software implementations that were created in the course of this thesis, target the application layer. The developed software package was implemented using the MATLAB scripting language and is called Multi-Modal Multi-Channel Joint Analysis Operator Learning (M3C-JAOL). In the following, its architecture is explained in more detail, since it determines the overall structure of the software programs.

6.2 Software architecture

The application layer defines data structures, implements data flow control and orchestrates the various processing tasks. All experiments in this thesis share a common pattern of task types which are executed in a processing chain. In all presented cases, the goal is to

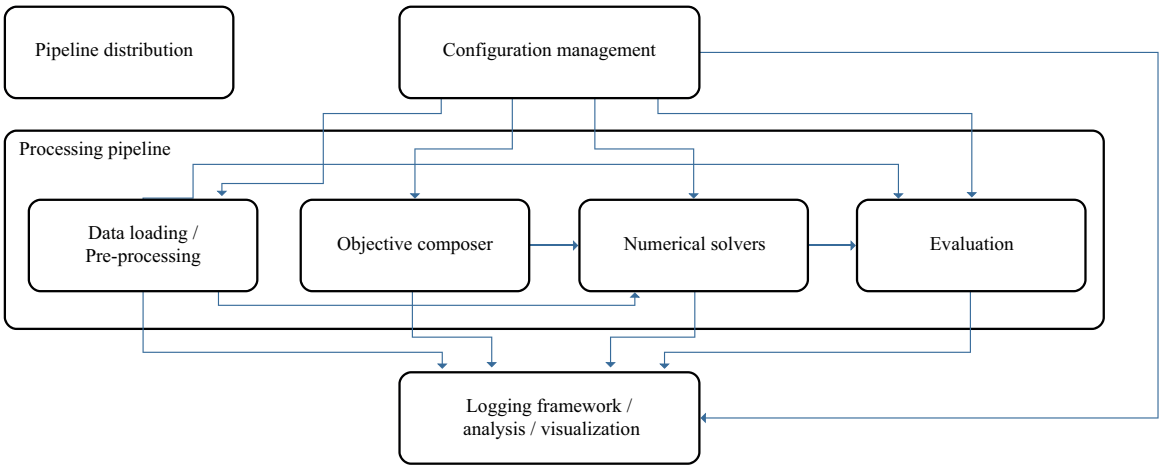


Figure 6.2: Architecture of the software implementation. Depicted is the separation of responsibilities among the main components and their interactions in the processing pipeline.

learn model parameters from training data or to apply a pre-trained model to reconstruct or segment images. A common structure can be observed, which involves the following task groups that interact with each other as depicted in Figure 6.2.

Data loading and pre-processing

All experiment runs start from loading test or training data. The data is stored in different formats and requires different types of pre-processing before it can be used in experiments. With the goal of providing correct input data for optimization and evaluation, this component is responsible for three task groups. First, it implements different methods to load experiment data from different locations, such as single files, nested directory structures, archives or databases. Second, it handles different file formats that represent the experiment data. In the presented experiments, single and multi-channel images, depth sensor data or MATLAB data is used and each of these formats needs to be handled differently in order to be represented on a uniform interface to other components. The third task group is the pre-processing of data for use in experiments. Typical sub-tasks of pre-processing include scale and offset normalizations of the data, recombination, tiling of the data into smaller parts (e.g. extraction of patches) or more complex operations such as projection of pixels from one image into the other (for instance required for multi-camera setups). This

component is designed such that it can be configured to run multiple processing steps sequentially and implements the necessary interfaces such that one step can be chained with another.

Objective composer

Among all presented methods, each comprises an objective function which is subsequently solved numerically. The corresponding optimization problems for learning, reconstruction or segmentation are constructed as weighted superpositions of multiple functions. To find solutions to these problems numerically, the solvers require an implementation for computing the objective value, given an estimate of the target variables, and directional derivatives of these functions with respect to the target variables (only first-order solvers are employed in this work and hence, no higher-order derivatives are required). The responsibility of the *objective composer* is to provide an implementation library of (sub-)functions and their directional derivatives, which are used to create compound objective functions. Based on a given configuration, it assembles the objective function and its gradient from parameterized library components and keeps track of the sub-results they yield during optimization.

Arguably one of the most difficult parts in engineering gradient based optimization algorithms, is the correct implementation of efficient directional derivatives, typically obtained by symbolic differentiation and vectorization. Transferring the derivatives to high-performance code is error-prone. To prevent flaws in this crucial step, an assisting method is built into the objective composer, commonly referred to as *gradient check*. It evaluates the implemented directional derivative $\frac{\partial f(\boldsymbol{X})}{\partial \boldsymbol{X}}$ of a function f at a random point $\boldsymbol{X}$ of the problem domain and compares it to a numerical differentiation at that point, technically asserting

$$\frac{\partial f(\boldsymbol{X})}{\partial \boldsymbol{X}} = \lim_{\boldsymbol{H} \to \boldsymbol{0}} \frac{f(\boldsymbol{X} + \boldsymbol{H}) - f(\boldsymbol{X})}{\boldsymbol{H}}. \tag{6.1}$$

By testing equality at multiple different random points $\boldsymbol{X}$ and over varying small values of $\boldsymbol{H}$, the correctness of the symbolic differentiation and its implementation are verified empirically (assuming sufficient smoothness of the function). The implementation of the gradient check in the objective composer can be run on single functions as well as weighted linear combinations of multiple functions and has proven to be a valuable tool for catching cod-

ing errors during development. The objective composer encapsulates parameters, function and gradient method handles of the optimization problem, and provides them in a unified interface for implementations in the numerical solver component.

Numerical solvers

Given the formalized optimization objective, the responsibility of this component is to provide methods for numerically finding its minimizers. It interfaces with the unified description of the objective function, provided by the objective composer and keeps track of optimization progress in each iteration. At the time of writing, it implemented the conjugate gradient and geometric conjugate gradient solvers (Chapters 3-5) as well as the ADMM solver of the Pottslab toolbox (Chapter 5).

Evaluation

In a subset of presented experiments, the objective function value provides all the necessary information to evaluate the success of the processing task. However, in the experiments where a trained model is used to solve a reconstruction or segmentation task, the objective function value alone is not sufficient to assess performance. In fact, other metrics are needed to judge the quality of the processing results. In the presented experiments, these metrics are all based on a comparison of the ground truth image or segmentation map with the output of the numerical solver. This type of evaluation is independent of the optimization problem and requires its own set of algorithm implementations to quantify success. In addition to consuming the result of the numerical solver, it makes use of data from the data loading and pre-processing component and is therefore designed as an independent component in the architecture. As all other components, it collects its own logging data of the evaluations for reporting.

Processing pipeline

The processing pipeline implements the software interfaces and component management facilities to instantiate and execute the different processing steps in the correct order and to control the data flow between the different components.

Configuration management

Experiments that quantitatively evaluate the effectiveness of a method typically need to execute the processing pipeline many times with varying input parameters. To avoid side-effects of varying implementations on results across different runs of an experiment, it is important to keep the code fixed and only change the configuration for different runs. Since the number of settings involved in arranging all components correctly for an experiment can quickly grow to hundreds of parameters, a component for managing configuration is implemented. It allows to define structured configuration objects, which segregate parameters for different parts of the application and which are then distributed to each of the components at startup. It supports versioning of configurations in structured files for later reference in experiment evaluations.

Logging framework

With each run of an experiment, the involved components perform crucial steps in the transformation of data to obtain a result. During development as well as during experiment execution, collecting detailed measurements of all steps is crucial for finding and addressing issues, selecting hyper-parameters and optimizing execution speed of the experiment. These measurement logs not only contain the data of a certain state in the processing chain but also information about execution times and iterations, which increases the dimensionality of this data further. As displaying all collected data delays execution and overloads the researcher, collection, analysis and presentation of these measurements need to be separated and visualization of this data at runtime needs to be configurable by the researcher. To address this holistically, a logging framework is implemented that defines a uniform data structure and

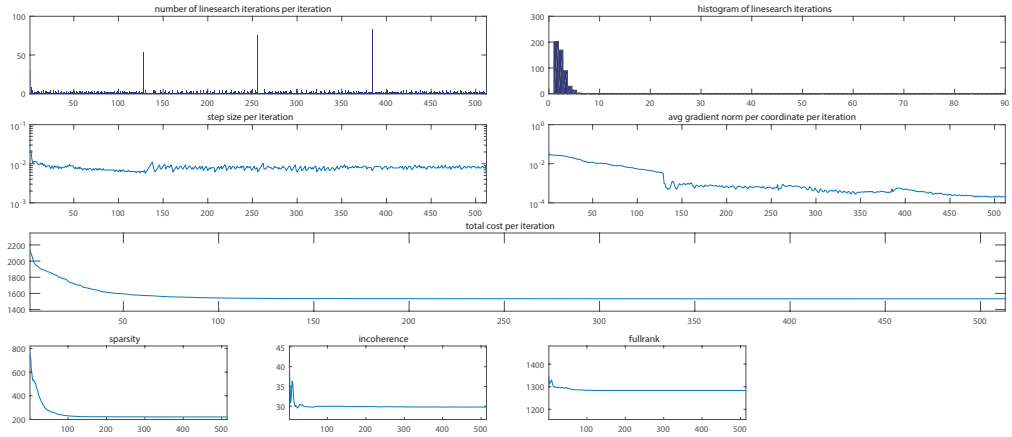


Figure 6.3: Visualization of logging data of the numerical solver component that reveals important metric progression across iterations of the optimization algorithm. This log dashboard shows the trend of objective function values and its sub-functions as well as statistics of the solver execution, including linesearch iterations, step sizes and gradient norm.

interfaces for all components to attach their logging data at runtime. Additional support for storing, analyzing and visualizing the logging data is also available. See Figure 6.3 for an example of how the logging component presents logging data of the numerical solver in a dashboard.

Pipeline distribution

Many algorithms require a suitable selection of hyper-parameters and the algorithms presented in this thesis are no exception. The exact interactions between the hyper-parameters and the data or the sought result are not always clear. Typically, suitable hyper-parameters are found empirically, demanding recurrent execution of the experiment with different settings to find the most suitable set of parameters. This is achieved by a grid search strategy that sweeps over a range of values of the parameters and this search can be time-consuming. However, since different runs of an experiment are independent of each other, they can be executed in parallel. The pipeline distribution component automates the splitting of configuration sets with different parameters, the execution of multiple pipelines in parallel and the collection of the respective results. In this way, the same version of the application

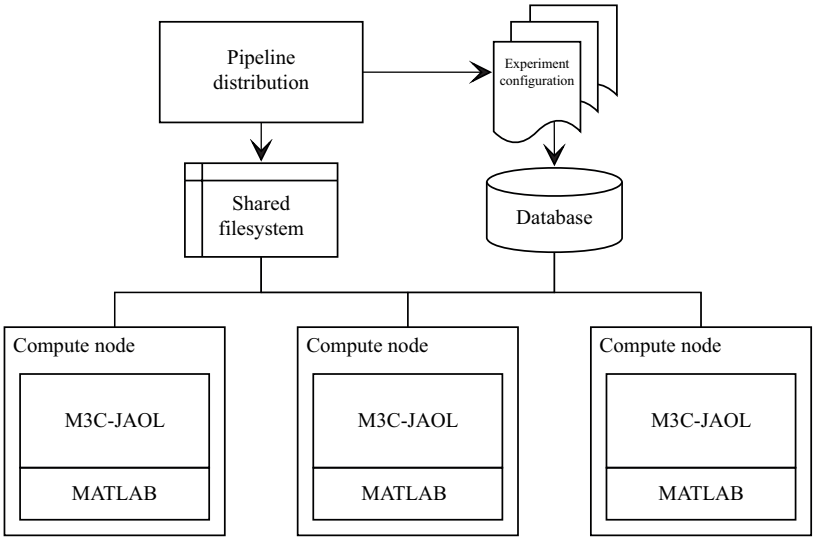


Figure 6.4: Schematic of pipeline distribution. Experiment parameterization are split into individual configurations and distributed across different compute nodes for parallel execution.

can be distributed to several computing resources and then parameterized by the pipeline distribution component. Figure 6.4 illustrates this process.

6.3 Discussion

In this chapter, the software implementation that allows the experimental evaluation of the methods presented in previous chapters was described. The engineering effort addresses the key requirements of quantitative algorithm evaluation and has been paramount in the work for this thesis and its related publications. This framework was revised several times during the work for this thesis, subsequently adopting best practices from the field and drawing heavily inspiration from several other scientific software packages, in particular from Caffe [86], Manopt [21] and TensorFlow [9].

In recent years, research communities in the data sciences have benefited to a great extend from the availability of better software frameworks for data modeling and numerical optimization. Several research institutes as well as large industry players have made their frameworks available to the public. Significant engineering efforts have helped to reduce the

time required by researches to transfer their models to software and make their execution efficient. In particular, parallel execution, distributed and GPU computing have become critical technologies to design larger models and parameterize them optimally from training data.

With this recent development, it should certainly be questioned if the effort required to build a custom framework is justified. However, many of the popular frameworks are designed for experimentation with model architectures and are less suited to implementation of custom solvers for unsupervised settings. One may therefore argue that in this case building a small and well tailored framework is valid. A shortcoming of the presented framework is certainly its lack of extensive support for GPU and distributed computing technologies. Future versions of this framework therefore need to extend these capabilities or build on top of the corresponding features of existing frameworks.

Chapter 7

Conclusion

In this thesis, model-based learning of low-level image features without supervision and their applications in image reconstruction, alignment, and segmentation were investigated. Several novel signal models were proposed that combine co-sparsity of the representations with knowledge of the image formation in these applications, along with algorithms to determine model parameters from training data. Efficient software implementations of these algorithms were developed to find numerical results for each of the proposed methods. In experimental evaluations with real data and standard benchmarks, these implementations were utilized to validate the ideas of each approach and to quantify their usefulness in practical image processing and computer vision challenges. State-of-the-art results were achieved in each of them, and in some cases prior art was even outperformed. Concretely, three new co-sparse analysis models for image data were proposed.

First, a co-sparse analysis model with centered rows was introduced. It addresses a shortcoming of previous methods for modeling photometric image data, which is typically brightness and contrast normalized for learning. These normalizations induce trivial solutions in the formulations of prior analysis operator learning approaches and it was shown how these are effectively avoided by learning centered analysis operators. A practical algorithm C-GOAL was designed and implemented that allows to learn centered analysis operators from real data on a sub-manifold of the oblique manifold. Experimental results suggested that this method required fewer training samples for learning and led to better analysis operators for use in reconstruction tasks than in previous approaches where the additional information of the illumination normalization was not considered.

Second, a co-sparse analysis model for bimodal image data was proposed. It is aimed at representing local image structures of spatially aligned images obtained by different imaging technologies through jointly sparse representations. With the assumption that an object which is captured by different imaging devices, such as photometric, infrared or depth cameras causes interdependent patterns in the respective recordings, the described Joint Bimodal Analysis Operator (JBAO) algorithm finds correlated sparse representations of spatially aligned bimodal image structures from real datasets. Subsequently, the learned model was exploited in bimodal sensor fusion applications. An algorithm was designed which makes use of a pre-trained bimodal analysis operator to regularize a highly ill-posed bimodal image reconstruction task. The algorithm was used to reconstruct a corrupted, noisy and low-resolution image by using a second high-quality image of a different modality to simultaneously denoise, inpaint and super-resolve the first. Experiments on photometric and depth datasets yielded state-of-the-art results. In addition, rigid image registration was tackled. There, pairs of images from different cameras which were misaligned through rigid transformations were automatically spatially registered by a new algorithm, which utilizes the dense analysis features from the pre-trained model in an optimization scheme on Lie groups. Experiments on intensity and depth as well as intensity and near infrared data yielded results that were competitive with the state-of-the-art.

The third model proposed in this thesis, targets the application of unsupervised texture segmentation. To this end, a spatial regularity condition was introduced into the analysis operator learning framework, which encourages the selection of image features that are approximately spatially piecewise constant. The designed algorithm was shown to learn useful texture features from unlabeled training data. To validate this model, a practical texture image segmentation algorithm based on the piecewise constant Mumford-Shah model was derived. Its experimental evaluation on standard unsupervised texture segmentation benchmarks generated state-of-the-art results, even outperforming most prior methods. In addition, the method's flexibility to effectively model entirely different types of texture data was evaluated positively through a segmentation experiment with histological images.

The discussed learning problems gave rise to mathematical functions which are very difficult to optimize numerically. To solve these problems, two important measures were required. First, non-smooth constraints were addressed with suitable relaxations, leading to

non-convex but smooth objective functions. Second, minimizers of these functions were found efficiently by designing geometric conjugate gradient solvers that act on Riemannian manifolds and exploit the intrinsic geometric structure of the problems. Conducted experiments support that both of these measures were effectively used to favorably trade accuracy with computability.

To conclude, the presented results provide further evidence that unsupervised learning of shallow representations with sparsity is widely applicable and useful for modeling image features. Furthermore, it is demonstrated that if additional knowledge about the image formation in an application is available, incorporating it into the learning task helps to improve model accuracy and to reduce the required number of training samples.

Publications

[1] M. Kiechle, M. Storath, A. Weinmann, and M. Kleinsteuber. "Model-Based Learning of Local Image Features for Unsupervised Texture Segmentation". In: *IEEE Transactions on Image Processing* 27.4 (Apr. 2018), pp. 1994–2007.

[2] D. Bobkov, M. Kiechle, S. Hilsenbeck, and E. Steinbach. "Room segmentation in 3D point clouds using anisotropic potential fields". In: *IEEE International Conference on Multimedia and Expo (ICME)*. July 2017, pp. 727–732.

[3] D. Bobkov et al. "Noise-resistant Unsupervised Object Segmentation in Multi-view Indoor Point Clouds". In: *12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISAPP)*. Mar. 2017.

[4] M. Kiechle, T. Habigt, S. Hawe, and M. Kleinsteuber. "A Bimodal Co-sparse Analysis Model for Image Processing". In: *International Journal of Computer Vision* 114.2 (2015), pp. 233–247.

[5] M. Kiechle, S. Hawe, and M. Kleinsteuber. "A Joint Intensity and Depth Co-Sparse Analysis Model for Depth Map Super-Resolution". In: *Proc. International Conference on Computer Vision*. 2013, pp. 1545–1552.

[6] K. Moser, M. Kiechle, and K. Ryokai. "Photocation: Tangible Learning System for DSLR Photography". In: *CHI '12 Extended Abstracts on Human Factors in Computing Systems*. 6. 2012, pp. 1691–1696.

[7] M. Hofmann, M. Kiechle, and G. Rigoll. "Late fusion for person detection in camera networks". In: *CVPR 2011 Workshops*. 2011, pp. 41–46.

[8] B. Römer, C. Menkens, J. Sußmann, and N. Konrad, eds. *Smart Grid Infrastructures - Trend Report 2010/2011*. Munich: Center for Digital Technology and Management (CDTM), 2011.

Bibliography

[9] M. Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. URL: `http://tensorflow.org/`.

[10] P. A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

[11] P.-A. Absil and K. A. Gallivan. "Joint diagonalization on the oblique manifold for independent component analysis". In: *2006 IEEE International Conference on Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings.* 5 (2006), pp. 945–948.

[12] M. Aharon, M. Elad, and A. Bruckstein. "K-SVD : An Algorithm for Designing of Overcomplete Dictionaries for Sparse Representation". In: *IEEE Transactions on Signal Processing* 54.11 (2006), pp. 4311–4322.

[13] G. Aresta et al. *BACH: Grand Challenge on Breast Cancer Histology Images*. 2018. arXiv: `1808.04277`.

[14] R. Azencott, J.-P. Wang, and L. Younes. "Texture classification using windowed Fourier filters". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19.2 (1997), pp. 148–153.

[15] S. Baker and T. Kanade. "Limits on super-resolution and how to break them". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.9 (Sept. 2002), pp. 1167–1183.

[16] A. Barron, J. Rissanen, and B. Yu. "The minimum description length principle in coding and modeling". In: *IEEE Transactions on Information Theory* 44.6 (1998), pp. 2743–2760.

[17] F. Bernardini et al. "The ball-pivoting algorithm for surface reconstruction". In: *IEEE Transactions on Visualization and Computer Graphics* 5.4 (1999), pp. 349–359.

[18] M. Bertero and P. Boccacci. *Introduction to inverse problems in imaging*. CRC press, 1998.

[19] A. V. Bhavsar and A. N. Rajagopalan. "Range Map Superresolution-Inpainting, and Reconstruction from Sparse Data". In: *Computer Vision and Image Understanding* 116.4 (2012), pp. 572–591.

[20] T. Blumensath and M. E. Davies. "Iterative hard thresholding for compressed sensing". In: *Applied and Computational Harmonic Analysis* 27.3 (2009), pp. 265–274.

[21] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. "Manopt, a Matlab Toolbox for Optimization on Manifolds". In: *Journal of Machine Learning Research* 15 (2014), pp. 1455–1459. URL: `http://www.manopt.org`.

[22] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[23] S. Boyd et al. "Distributed optimization and statistical learning via the alternating direction method of multipliers". In: *Foundations and Trends® in Machine Learning* 3.1 (2011), pp. 1–122.

[24] L. G. Brown. "A survey of image registration techniques". In: *ACM Computing Surveys (CSUR)* 24 (1992), pp. 325–376.

[25] M. Brown and S. Süsstrunk. "Multi-spectral SIFT for scene category recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2011, pp. 177–184.

[26] J.-F. Cai, S. Osher, and Z. Shen. "Split Bregman Methods and Frame Based Image Restoration". In: *Multiscale Modeling & Simulation* 8.2 (Jan. 2010), pp. 337–369.

[27] G. Calvert, C. Spence, and B. Stein. *The handbook of multisensory processes*. 2004.

[28] E. J. Candès and L. Demanet. "The curvelet representation of wave propagators is optimally sparse". In: *Communications on Pure and Applied Mathematics* 58.11 (Nov. 2005), pp. 1472–1528.

[29] E. J. Candès, Y. C. Eldar, D. Needell, and P. Randall. "Compressed sensing with coherent and redundant dictionaries". In: *Applied and Computational Harmonic Analysis* 31.1 (2011), pp. 59–73.

[30] E. J. Candes and T. Tao. "Near-Optimal Signal Recovery From Random Projections: Universal Encoding Strategies?" In: *IEEE Transactions on Information Theory* 52.12 (Dec. 2006), pp. 5406–5425.

[31] E. J. Candès, M. B. Wakin, and S. P. Boyd. "Enhancing Sparsity by Reweighted ℓ_1 Minimization". In: *Journal of Fourier Analysis and Applications* 14.5-6 (Oct. 2008), pp. 877–905.

[32] A. Chambolle. "Finite-differences discretizations of the Mumford-Shah functional". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 33.02 (1999), pp. 261–288.

[33] D. Chan, H. Buisman, C. Theobalt, and S. Thrun. "A Noise-Aware Filter for Real-Time Depth Upsampling". In: *Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications*. 2008.

[34] T. Chan and J. Shen. *Image Processing and Analysis*. Society for Industrial and Applied Mathematics, 2005.

[35] R. Chartrand. "Exact Reconstruction of Sparse Signals via Nonconvex Minimization". In: *IEEE Signal Processing Letters* 14.10 (Oct. 2007), pp. 707–710.

[36] R. Chartrand. "Fast algorithms for nonconvex compressive sensing: MRI reconstruction from very few data". In: *Biomedical Imaging: From Nano to Macro, 2009. ISBI'09. IEEE International Symposium on*. IEEE. 2009, pp. 262–265.

[37] Y. Chen, R. Ranftl, and T. Pock. "Insights Into Analysis Operator Learning: From Patch-Based Sparse Models to Higher Order MRFs." In: *IEEE Transactions on Image Processing* 23.3 (Mar. 2014), pp. 1060–72.

[38] X. Cheng, M. Zeng, and X. Liu. "Feature-preserving filtering with L_0 gradient minimization". In: *Computers & Graphics* 38 (2014), pp. 150–157.

[39] A. A. Cole-Rhodes, K. L. Johnson, J. LeMoigne, and I. Zavorin. "Multiresolution registration of remote sensing imagery by optimization of mutual information using a stochastic gradient." In: *IEEE Transactions on Image Processing* 12.12 (Jan. 2003), pp. 1495–511.

[40] A. Collignon et al. "Automated multi-modality image registration based on information theory". In: *Information processing in medical imaging*. Vol. 3. 1995, pp. 263–274.

[41] W. Commons. *The Microsoft Kinect peripheral for the Xbox 360*. 2011. URL: %7Bhttps://upload.wikimedia.org/wikipedia/commons/6/67/Xbox-360-Kinect-Standalone.png%7D.

[42] S. Cotter, B. Rao, and K. Kreutz-Delgado. "Sparse solutions to linear inverse problems with multiple measurement vectors". In: *IEEE Transactions on Signal Processing* 53.7 (July 2005), pp. 2477–2488.

[43] W. Dai and O. Milenkovic. "Subspace pursuit for compressive sensing signal reconstruction". In: *IEEE Transactions on Information Theory* 55.5 (2009), pp. 2230–2249.

[44] Y. Dai and Y. Yuan. "An Efficient Hybrid Conjugate Gradient Method for Unconstrained Optimization". In: *Annals of Operations Research* 103.1-4 (2001), pp. 33–47.

[45] G. Davis, S. Mallat, and M. Avellaneda. "Adaptive greedy approximations". In: *Constructive Approximation* 13.1 (Mar. 1997), pp. 57–98.

[46] L. Demanet and L. Ying. "Wave atoms and sparsity of oscillatory patterns". In: *Applied and Computational Harmonic Analysis* 23.3 (Nov. 2007), pp. 368–387.

[47] J. Diebel and S. Thrun. "An Application of Markov Random Fields to Range Sensing". In: *NIPS*. Vol. 18. 2005, pp. 291–298.

[48] J. Dolson, J. Baek, C. Plagemann, and S. Thrun. "Upsampling Range Data in Dynamic Environments". In: *CVPR*. 2010, pp. 1141–1148.

[49] D. L. Donoho. "De-Noising by Soft-Thresholding". In: *IEEE Transactions on Information Theory* 41.3 (May 1995), pp. 613–627. arXiv: 0611061v2 [arXiv:quant-ph].

[50] D. L. Donoho and M. Elad. "Optimally sparse representation in general (nonorthogonal) dictionaries via l minimization." In: *Proceedings of the National Academy of Sciences of the United States of America* 100.5 (Mar. 2003), pp. 2197–2202.

[51] M. Duarte et al. "Distributed Compressed Sensing of Jointly Sparse Signals". In: *Asilomar Conference on Signals, Systems and Computers*. IEEE, 2005, pp. 1537–1541. arXiv: 0901.3403v1.

[52] J. Eckstein and D. P. Bertsekas. "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators". In: *Mathematical Programming* 55.1-3 (Apr. 1992), pp. 293–318. arXiv: 1306.3203.

[53] M. Elad, P. Milanfar, and R. Rubinstein. "Analysis versus Synthesis in Signal Priors". In: *Inverse Problems* 23.3 (2007), pp. 947–968.

[54] M. Elad. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer Publishing Company, Incorporated, 2010.

[55] M. Elad, M. A. T. Figueiredo, and Yi Ma. "On the Role of Sparse and Redundant Representations in Image Processing". In: *Proceedings of the IEEE* 98.6 (June 2010), pp. 972–982.

[56] X. Fan, H. Rhody, and E. Saber. "A Spatial-Feature-Enhanced MMI Algorithm for Multimodal Airborne Image Registration". In: *IEEE Transactions on Geoscience and Remote Sensing* 48.6 (June 2010), pp. 2580–2589.

[57] R. Fattal. "Image Upsampling via Imposed Edge Statistics". In: *ACM Transactions on Graphics* 26.3 (2007).

[58] S. Foucart. "Hard Thresholding Pursuit: An Algorithm for Compressive Sensing". In: *SIAM Journal on Numerical Analysis* 49.6 (Jan. 2011), pp. 2543–2563.

[59] G. Freedman and R. Fattal. "Image and Video Upscaling from Local Self-Examples". In: *ACM Transactions on Graphics* 30.2 (2011), pp. 1–11.

[60] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. "Learning Low-Level Vision". In: *International Journal of Computer Vision* 40.1 (Oct. 2000), pp. 25–47.

[61] M. Galun, E. Sharon, R. Basri, and A. Brandt. "Texture segmentation by multiscale aggregation of filter responses and shape elements". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2003, pp. 716–723.

[62] S. Geman and D. Geman. "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6.6 (1984), pp. 721–741.

[63] R. Giryes and M. Elad. "Cosamp and SP for the cosparse analysis model". In: *2012 Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*. Aug. 2012, pp. 964–968.

[64] R. Giryes, S. Nam, R. Gribonval, and M. E. Davies. "Iterative cosparse projection algorithms for the recovery of cosparse vectors". In: *2011 19th European Signal Processing Conference*. Aug. 2011, pp. 1460–1464.

[65] R. Giryes et al. "Greedy-like algorithms for the cosparse analysis model". In: *Linear Algebra and Its Applications* 441 (2014), pp. 22–60. arXiv: 1207.2456.

[66] Y. Grandvalet and S. Canu. "Outcomes of the equivalence of adaptive ridge with least absolute shrinkage". In: *Advances in Neural Information Processing Systems* (1998), pp. 445–451.

[67] S. A. Gudmundsson and J. R. Sveinsson. "ToF-CCD Image Fusion using Complex Wavelets". In: *ICASSP*. 2011, pp. 1557–1560.

[68] M. Haindl and S. Mikes. "Unsupervised Texture Segmentation Using Multispectral Modelling Approach". In: *IEEE International Conference on Pattern Recognition (ICPR)*. Vol. 2. 2006, pp. 203–206.

[69] M. Haindl and S. Mikes. "Texture Segmentation Benchmark". In: *Proceedings of the IEEE International Conference on Pattern Recognition*. 2008, pp. 1–4.

[70] M. Haindl and S. Mikes. *The Prague Texture Segmentation Datagenerator and Benchmark*. http://mosaic.utia.cas.cz. accessed 2016-06-13. 2016.

[71] M. Haindl and S. Mikeš. "Model-Based Texture Segmentation". In: Springer Berlin Heidelberg, 2004, pp. 306–313.

[72] P. Hall, G. Kerkyacharian, and D. Picard. "On the minimax optimality of block thresholded wavelet estimators". In: *Statistica Sinica* (1999), pp. 33–49.

[73] S. Hawe, M. Kleinsteuber, and K. Diepold. "Cartoon-like Image Reconstruction via Constrained lp-Minimization". In: *ICASSP*. 2012, pp. 717–720.

[74] S. Hawe, M. Kleinsteuber, and K. Diepold. "Analysis Operator Learning and Its Application to Image Reconstruction". In: *IEEE Transactions on Image Processing* 22.6 (2013), pp. 2138–2150.

[75] S. Hawe. "Learning Sparse Data Models via Geometric Optimization with Applications to Image Processing". PhD thesis. Technische Universität München, 2013, p. 131. URL: http://mediatum.ub.tum.de/doc/1154199/1154199.pdf.

[76] S. Hawe, M. Seibert, and M. Kleinsteuber. "Separable Dictionary Learning". In: *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. 2013, pp. 438–445.

[77] M. Hestenes and E. Stiefel. "Methods of conjugate gradients for solving linear systems". In: *Journal of Research of the National Bureau of Standards* 49.6 (1952), p. 409. arXiv: 1102.0183.

[78] G. E. Hinton and R. R. Salakhutdinov. "Reducing the Dimensionality of Data with Neural Networks". In: *Science* 313.5786 (2006), pp. 504–507.

[79] Hong Chang, Dit-Yan Yeung, and Yimin Xiong. "Super-resolution through neighbor embedding". In: *Computer Vision and Pattern Recognition.* Vol. 1. 2004, pp. 275–282.

[80] J. Huang and D. Mumford. "Statistics of natural images and models". In: *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149).* 1999, pp. 541–547.

[81] J. Huang and T. Zhang. "The benefit of group sparsity". In: *Ann. Statist.* 38.4 (Aug. 2010), pp. 1978–2004.

[82] M. Hyder and K. Mahata. "A Robust Algorithm for Joint-Sparse Recovery". In: *IEEE Signal Processing Letters* 16.12 (Dec. 2009), pp. 1091–1094.

[83] A. Jain and F. Farrokhnia. "Unsupervised texture segmentation using Gabor filters". In: *Pattern Recognition* 24.12 (1991), pp. 1167–1186.

[84] R. Jenatton, J.-Y. Audibert, and F. Bach. "Structured variable selection with sparsity-inducing norms". In: *Journal of Machine Learning Research* 12.Oct (2011), pp. 2777–2824.

[85] K. Jia, X. Wang, and X. Tang. "Image transformation based on learning dictionaries across image spaces." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.2 (Feb. 2013), pp. 367–80.

[86] Y. Jia et al. "Caffe: Convolutional Architecture for Fast Feature Embedding". In: *arXiv preprint arXiv:1408.5093* (2014).

[87] Z. Kato and T.-C. Pong. "A Markov random field image segmentation model for color textured images". In: *Image and Vision Computing* 24.10 (2006), pp. 1103–1114.

[88] J. B. Keller. "Inverse Problems". In: *The American Mathematical Monthly* 83.2 (1976), pp. 107–118.

[89] K. Khoshelham and S. O. Elberink. "Accuracy and resolution of Kinect depth data for indoor mapping applications". In: *Sensors* 12.2 (2012), pp. 1437–1454.

[90] S. Kitić, L. Albera, N. Bertin, and R. Gribonval. "Physics-Driven Inverse Problems Made Tractable With Cosparse Regularization". In: *IEEE Transactions on Signal Processing* 64.2 (Jan. 2016), pp. 335–348.

[91] S. Kitić, N. Bertin, and R. Gribonval. "A review of cosparse signal recovery methods applied to sound source localization". In: *Le XXIVe colloque Gretsi*. Brest, France, Sept. 2013.

[92] S. Klein et al. "elastix: A Toolbox for Intensity-Based Medical Image Registration." In: *IEEE Transactions on Medical Imaging* 29.1 (Jan. 2010), pp. 196–205.

[93] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele. "Joint Bilateral Upsampling". In: *ACM Transactions on Graphics* 26.3 (2007).

[94] S. J. Krotosky and M. M. Trivedi. "Mutual information based registration of multimodal stereo videos for person tracking". In: *Computer Vision and Image Understanding* 106.2-3 (May 2007), pp. 270–287.

[95] M. Li and P. Vitányi. *An introduction to Kolmogorov complexity and its applications*. Springer Science & Business Media, 2013.

[96] Y. Li, T. Xue, L. Sun, and J. Liu. "Joint Example-Based Depth Map Super-Resolution". In: *IEEE International Conference on Multimedia and Expo*. 2012, pp. 152–157.

[97] C. Liu, H.-Y. Shum, and W. T. Freeman. "Face Hallucination: Theory and Practice". In: *International Journal of Computer Vision* 75.1 (Feb. 2007), pp. 115–134.

[98] X. Liu and D. Wang. "Image and texture segmentation using local spectral histograms". In: *IEEE Transactions on Image Processing* 15.10 (2006), pp. 3066–3077.

[99] J. Long, E. Shelhamer, and T. Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3431–3440.

[100] J. Lu, D. Min, R. S. Pahwa, and M. N. Do. "A Revisit to MRF-Based Depth Map Super-Resolution and Enhancement". In: *ICASSP*. 2011, pp. 985–988.

[101] Y. Lu and M. Do. "A Theory for Sampling Signals From a Union of Subspaces". In: *IEEE Transactions on Signal Processing* 56.6 (June 2008), pp. 2334–2345.

[102] O. Mac Aodha, N. D. Campbell, A. Nair, and G. J. Brostow. "Patch based synthesis for single depth image super-resolution". In: *European Conference on Computer Vision*. 2012, pp. 71–84.

[103] J. MacQueen et al. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

[104] M. Mahmoudi and G. Sapiro. "Sparse Representations for Range Data Restoration". In: *IEEE Transactions on Image Processing* 21.5 (2012), pp. 2909–2915.

[105] J. Mairal, M. Elad, and G. Sapiro. "Sparse Representation for Color Image Restoration". In: *IEEE Transactions on Image Processing* 17.1 (2008), pp. 53–69.

[106] J. Mairal, F. Bach, and J. Ponce. "Task-driven dictionary learning." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.4 (Apr. 2012), pp. 791–804.

[107] J. Mairal, F. Bach, and J. Ponce. "Sparse Modeling for Image and Vision Processing". In: *Foundations and Trends® in Computer Graphics and Vision* 8.2-3 (2014), pp. 85–283.

[108] S. G. Mallat. "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 11 (1989), pp. 674–693.

[109] S. Mallat and Zhifeng Zhang. "Matching pursuits with time-frequency dictionaries". In: *IEEE Transactions on Signal Processing* 41.12 (1993), pp. 3397–3415.

[110] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 1999, pp. 20–41.

[111] S. Mallat. *A wavelet tour of signal processing: the sparse way*. 3rd. Academic Press, 2008.

[112] M. W. Marcellin, M. J. Gormish, A. Bilgin, and M. P. Boliek. "An overview of JPEG-2000". In: *Proceedings DCC 2000. Data Compression Conference*. IEEE. 2000, pp. 523–541.

[113] D. Martin, C. Fowlkes, and J. Malik. "Learning to detect natural image boundaries using local brightness, color, and texture cues". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.5 (2004), pp. 530–549.

[114] D. Mattes et al. "PET-CT image registration in the chest using free-form deformations." In: *IEEE Transactions on Medical Imaging* 22.1 (Jan. 2003), pp. 120–128.

[115] M. McCann et al. "Images as occlusions of textures: a framework for segmentation". In: *IEEE Transactions on Image Processing* 23.5 (2014), pp. 2033–2046.

[116] H. McGurk and J. MacDonald. "Hearing lips and seeing voices". In: *Nature* (1976).

[117] N. Mevenkamp and B. Berkels. "Variational multi-phase segmentation using high-dimensional local features". In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2016, pp. 1–9.

[118] M. Mishali and Y. Eldar. "Reduce and Boost: Recovering Arbitrary Sets of Jointly Sparse Vectors". In: *IEEE Transactions on Signal Processing* 56.10 (Oct. 2008), pp. 4692–4702.

[119] H. Mobahi et al. "Segmentation of natural images by texture and boundary compression". In: *International Journal of Computer Vision* 95.1 (2011), pp. 86–98.

[120] D. Mumford and J. Shah. "Optimal approximations by piecewise smooth functions and associated variational problems". In: *Communications on Pure and Applied Mathematics* 42.5 (1989), pp. 577–685.

[121] S. Nam, M. Davies, M. Elad, and R. Gribonval. "The cosparse analysis model and algorithms". In: *Applied and Computational Harmonic Analysis* 34.1 (2013), pp. 30–56.

[122] S. Nam and R. Gribonval. "Physics-driven structured cosparse modeling for source localization". In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Mar. 2012, pp. 5397–5400.

[123] S. Nam, M. E. Davies, M. Elad, and R. Gribonval. "Cosparse analysis modeling - uniqueness and algorithms". In: *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Nov. 2011, pp. 5804–5807.

[124] P. K. Nathan Silberman Derek Hoiem and R. Fergus. "Indoor Segmentation and Support Inference from RGBD Images". In: *ECCV*. 2012.

[125] D. Needell and J. Tropp. "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples". In: *Applied and Computational Harmonic Analysis* 26.3 (May 2009), pp. 301–321.

[126] R. Nguyen and M. Brown. "Fast and Effective L_0 Gradient Minimization by Region Fusion". In: *IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 208–216.

[127] C. Nieuwenhuis, S. Hawe, M. Kleinsteuber, and D. Cremers. "Co-Sparse Textural Similarity for Interactive Segmentation". In: *ECCV 2014* (2014), pp. 285–301.

[128] J. Nocedal and S. Wright. *Numerical Optimization*. 1999, p. 651.

[129] G. Obozinski, B. Taskar, and M. I. Jordan. "Joint covariate selection and joint subspace selection for multiple classification problems". In: *Statistics and Computing* 20.2 (Apr. 2010), pp. 231–252.

[130] T. Ojala, M. Pietikainen, and T. Maenpaa. "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.7 (2002), pp. 971–987.

[131] B. A. Olshausen and D. J. Field. "Emergence of simple-cell receptive field properties by learning a sparse code for natural images". In: *Nature* 381.6583 (1996), p. 607.

[132] G. Ongie and M. Jacob. "Recovery of Discontinuous Signals Using Group Sparse Higher Degree Total Variation". In: *IEEE Signal Processing Letters* 22.9 (Sept. 2015), pp. 1414–1418.

[133] B. Ophir, M. Elad, N. Bertin, and M. D. Plumbley. "Sequential Minimal Eigenvalues - An Approach to Analysis Dictionary Learning". In: *EUSIPCO*. 2011, pp. 1465–1469.

[134] J. Orchard. "Efficient least squares multimodal registration with a globally exhaustive alignment search." In: *IEEE Transactions on Image Processing* 16.10 (Oct. 2007), pp. 2526–2534.

[135] M. Ozden and E. Polat. "Image segmentation using color and texture features". In: *Proceedings of the 13th European Signal Processing Conference*. 2005, pp. 2226–2229.

[136] C. Panagiotakis, I. Grinias, and G. Tziritas. "Natural Image Segmentation Based on Tree Equipartition, Bayesian Flooding and Region Merging". In: *IEEE Transactions on Image Processing* 20.8 (2011), pp. 2276–2287.

[137] C. Panagiotakis, I. Grinias, and G. Tziritas. *Texture Segmentation Based on Voting of Blocks, Bayesian Flooding and Region Merging*. `https://sites.google.com/site/costaspanagiotakis/research/imagesegmentation`. accessed 2016-06-14. 2014.

[138] J. Park, H. Kim, M. S. Brown, and I. Kweon. "High Quality Depth Map Upsampling for 3D-TOF Cameras". In: *ICCV*. 2011, pp. 1623–1630.

[139] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad. "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition". In: *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*. Nov. 1993, 40–44 vol.1.

[140] K. F. Pearson. "On lines and planes of closest fit to systems of points in space". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572.

[141] Y. Peng et al. "RASL: robust alignment by sparse and low-rank decomposition for linearly correlated images." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.11 (Nov. 2012), pp. 2233–46.

[142] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever. "Mutual-information-based registration of medical images: a survey." In: *IEEE Transactions on Medical Imaging* 22.8 (Aug. 2003), pp. 986–1004.

[143] T. Randen and J. H. Husoy. "Filtering for texture classification: A comparative study". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21.4 (1999), pp. 291–310.

[144] M. Ranzato, C. Poultney, S. Chopra, and Y. LeCun. "Efficient Learning of Sparse Representations with an Energy-based Model". In: *Proceedings of the 19th International Conference on Neural Information Processing Systems*. NIPS'06. Canada: MIT Press, 2006, pp. 1137–1144.

[145] S. Ravishankar and Y. Bresler. "Learning Sparsifying Transforms". In: *IEEE Transactions on Signal Processing* 61.5 (Mar. 2013), pp. 1072–1086.

[146] X. Ren and J. Malik. "Learning a classification model for segmentation". In: *IEEE International Conference on Computer Vision*. 2003, pp. 10–17.

[147] M. Rousson, T. Brox, and R. Deriche. "Active unsupervised texture segmentation on a diffusion based feature space". In: *IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 2. Madison, 2003, pp. II–699.

[148] R. Rubinstein, T. Faktor, and M. Elad. "K-SVD dictionary-learning for the analysis sparse model". In: *IEEE International Conference on Acoustics, Speech and Signal Processing*. 2012, pp. 5405–5408.

[149] R. Rubinstein, A. M. Bruckstein, and M. Elad. "Dictionaries for sparse representation modeling". In: *Proceedings of the IEEE* 98.6 (2010), pp. 1045–1057.

[150] R. Rubinstein, T. Peleg, and M. Elad. "Analysis K-SVD: A Dictionary-Learning Algorithm for the Analysis Sparse Model". In: *IEEE Transactions on Signal Processing* 61.3 (Feb. 2013), pp. 661–677.

[151] L. I. Rudin, S. Osher, and E. Fatemi. "Nonlinear Total Variation Based Noise Removal Algorithms". In: *Physica D: Nonlinear Phenomena* 60.1 (1992), pp. 259–268.

[152] S. Sarvotham et al. "Distributed compressed sensing of jointly sparse signals". In: *Asilomar conference on signals, systems, and computers*. 2005, pp. 1537–1541.

[153] G. Scarpa, R. Gaetano, M. Haindl, and J. Zerubia. "Hierarchical Multiple Markov Chain Model for Unsupervised Texture Segmentation". In: *IEEE Transactions on Image Processing* 18.8 (2009), pp. 1830–1843.

[154] D. Scharstein and R. Szeliski. "High-Accuracy Stereo Depth Maps using Structured Light". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2003, pp. 195–202.

[155] S. Schuon, C. Theobalt, J. Davis, and S. Thrun. "LidarBoost: Depth Superresolution for ToF 3D Shape Scanning". In: *CVPR*. 2009, pp. 343–350.

[156] I. W. Selesnick and M. A. T. Figueiredo. "Signal Restoration with Overcomplete Wavelet Transforms: Comparison of Analysis and Synthesis Priors". In: *SPIE Optical Engineering and Applications*. 2009, pp. 74460D–74460D.

[157] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision*. 2007.

[158] M. Storath and A. Weinmann. "Fast partitioning of vector-valued images". In: *SIAM Journal on Imaging Sciences* 7.3 (2014), pp. 1826–1852.

[159] M. Storath, A. Weinmann, and L. Demaret. "Jump-sparse and sparse recovery using Potts functionals". In: *IEEE Transactions on Signal Processing* 62.14 (2014), pp. 3654–3666.

[160] M. Storath, A. Weinmann, and M. Unser. "Unsupervised texture segmentation using monogenic curvelets and the Potts model". In: *IEEE International Conference on Image Processing (ICIP)*. 2014, pp. 4348–4352.

[161] C. Studholme, D. Hill, and D. Hawkes. "An overlap invariant entropy measure of 3D medical image alignment". In: *Pattern Recognition* 32.1 (Jan. 1999), pp. 71–86.

[162] R. Szeliski. *Computer vision: Algorithms and applications*. Springer Science & Business Media, 2010.

[163] R. Tibshirani et al. "Sparsity and Smoothness via the Fused Lasso". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.1 (2005), pp. 91–108.

[164] S. Todorovic and N. Ahuja. "Texel-based texture segmentation". In: *IEEE International Conference on Computer Vision*. 2009, pp. 841–848.

[165] I. Tosic and S. Drewes. "Learning joint intensity-depth sparse representations". In: *IEEE Transactions on Image Processing* 23.5 (May 2014), pp. 2122–2132. arXiv: 1201.0566.

[166] J. A. Tropp. "Algorithms for simultaneous sparse approximation. Part II: Convex relaxation". In: *Signal Processing* 86.3 (2006), pp. 589–602.

[167] J. A. Tropp, A. C. Gilbert, and M. J. Strauss. "Algorithms for simultaneous sparse approximation. Part I: Greedy pursuit". In: *Signal Processing* 86.3 (Mar. 2006), pp. 572–588.

[168] B. A. Turlach, W. N. Venables, and S. J. Wright. "Simultaneous Variable Selection". In: *Technometrics* 47.3 (Aug. 2005), pp. 349–363.

[169] J. Uhrig et al. "Sparsity Invariant CNNs". In: *International Conference on 3D Vision (3DV)*. 2017.

[170] M. Unser. "Texture classification and segmentation using wavelet frames". In: *IEEE Transactions on Image Processing* 4.11 (1995), pp. 1549–1560.

[171] M. Unser and M. Eden. "Nonlinear operators for improving texture segmentation based on features extracted by spatial filtering". In: *IEEE Transactions on Systems, Man, and Cybernetics* 20.4 (1990), pp. 804–815.

[172] S. Vaiter, G. Peyre, C. Dossal, and J. Fadili. "Robust Sparse Analysis Regularization". In: *IEEE Transactions on Information Theory* 59.4 (Apr. 2013), pp. 2001–2016.

[173] P. Vincent et al. "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion Pierre-Antoine Manzagol". In: *Journal of Machine Learning Research* 11 (2010), pp. 3371–3408.

[174] P. Viola and W. M. Wells III. "Alignment by maximization of mutual information". In: *International Journal of Computer Vision* 24.2 (1997), pp. 137–154.

[175] S. Wang, D. Zhang, Y. Liang, and Q. Pan. "Semi-coupled dictionary learning with applications to image super-resolution and photo-sketch synthesis". In: *IEEE Conference on Computer Vision and Pattern Recognition*. June 2012, pp. 2216–2223.

[176] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. "Image quality assessment: from error visibility to structural similarity". In: *IEEE Transactions on Image Processing* 13.4 (Apr. 2004), pp. 600–612.

[177] B. Wen, S. Ravishankar, and Y. Bresler. "Structured Overcomplete Sparsifying Transform Learning with Convergence Guarantees and Applications". In: *International Journal of Computer Vision* 114.2-3 (Sept. 2015), pp. 137–167.

[178] J. Wörmann, S. Hawe, and M. Kleinsteuber. "Analysis Based Blind Compressive Sensing". In: *Signal Processing Letters, IEEE* 20.5 (2013), pp. 491–494.

[179] S. Wright and J. Nocedal. "Numerical optimization". In: *Springer Science* 35.67-68 (1999), p. 7.

[180] Y. Xia, D. Feng, and R. Zhao. "Morphology-based multifractal estimation for texture segmentation". In: *IEEE Transactions on Image Processing* 15.3 (2006), pp. 614–623.

[181] X. Xiang, G. Li, J. Tong, and Z. Pan. "Fast and Simple Super Resolution for Range Data". In: *International Conference on Cyberworlds*. 2010, pp. 319–324.

[182] L. Xu, C. Lu, Y. Xu, and J. Jia. "Image smoothing via L_0 gradient minimization". In: *ACM Transactions on Graphics (TOG)* 30.6 (2011), p. 174.

[183] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies. "Analysis Operator Learning for Overcomplete Cosparse Representations". In: *EUSIPCO*. 2011, pp. 1470–1474.

[184] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies. "Noise Aware Analysis Operator Learning for Approximately Cosparse Signals". In: *ICASSP*. 2012, pp. 5409–5412.

[185] M. Yaghoobi, S. Nam, R. Gribonval, and M. E. Davies. "Constrained Overcomplete Analysis Operator Learning for Cosparse Signal Modelling". In: *IEEE Transactions on Signal Processing* 61.9 (May 2013), pp. 2341–2355.

[186] J. Yang, J. Wright, T. Huang, and Y. Ma. "Image Super-Resolution via Sparse Representation." In: *IEEE Transactions on Image Processing* 19.11 (Nov. 2010), pp. 2861–2873.

[187] Q. Yang, R. Yang, J. Davis, and D. Nistér. "Spatial-Depth Super Resolution for Range Images". In: *CVPR*. 2007, pp. 1–8.

[188] J. Yuan, D. Wang, and A. Cheriyadat. "Factorization-based texture segmentation". In: *IEEE Transactions on Image Processing* 24.11 (2015), pp. 3488–3497.

[189] J. Yuan, D. Wang, and R. Li. "Image segmentation using local spectral histograms and linear regression". In: *Pattern Recognition Letters* 33.5 (2012), pp. 615–622.

[190] M. Yuan and Y. Lin. "Model selection and estimation in regression with grouped variables". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68.1 (2006), pp. 49–67.

[191] R. Zeyde, M. Elad, and M. Protter. "On single image scale-up using sparse-representations". In: *Curves and Surfaces* (2012).

[192] J. Zhu, L. Wang, R. Yang, and J. Davis. "Fusion of Time-of-Flight Depth and Stereo for High Accuracy Depth Maps". In: *CVPR*. 2008, pp. 1–8.

[193] B. Zitová and J. Flusser. "Image registration methods: a survey". In: *Image and Vision Computing* 21.11 (Oct. 2003), pp. 977–1000.

Appendix A: Optimization on matrix manifolds

The learning rule for obtaining analysis operators from data samples, posed in Eq. (3.17), is a constrained optimization problem. Constrained optimization is a well established research area and powerful techniques exist to address general problems in this area [179, 22]. In the optimization tasks presented in this thesis, a specific class of constraints is of particular interest. These constraints express that the solution of the problem is located on a manifold. Therefore, they are called geometric constraints and the methods designed for tackling problems with such constraints are called geometric optimization methods [10]. In comparison to classical optimization methods which operate in an embedding space that can be of much higher dimension than the manifold, geometric optimization works on the manifold directly, often leading to lower complexity and better numerical properties [10].

The problems discussed in this thesis are solved using unconstrained optimization in a constrained search space that resembles a high-dimensional sphere or a product of spheres caused by the requirement that iterates must be of fixed norm. What follows is a brief review of the first-order line-search method called geometric conjugate gradients. For a thorough introduction to the topic and detailed discussions of different geometric optimization algorithms and their convergence analyses, the reader is referred to the text book [10].

Line search methods on matrix manifolds

Let $\mathcal{M}$ be a smooth Riemannian sub-manifold of a finite dimensional real vector space $\mathbb{V}$ with a scalar product $\langle \cdot, \cdot \rangle$ and consider the problem of minimizing a smooth real valued

function

$$f\colon \mathcal{M} \to \mathbb{R}. \tag{A.1}$$

The general idea of line search methods like conjugate gradient or gradient descent algorithms on manifolds is that, starting from some point $\boldsymbol{O} \in \mathcal{M}$, the minimizer of Eq. (A.1) is searched along a curve on the manifold. In this setting, the descent direction is an element of the tangent space $T_{\boldsymbol{O}}\mathcal{M}$ at $\boldsymbol{O}$, and the updated iterate is searched along geodesics of the manifold. In the case where f is defined in the embedding space $\mathbb{V}$, its gradient $\nabla f(\boldsymbol{O})$ with respect to $\langle \cdot, \cdot \rangle$ is uniquely determined by

$$\tfrac{\mathrm{d}}{\mathrm{d}t}\big|_{t=0} f(\boldsymbol{O} + t\boldsymbol{H}) = \langle \nabla f(\boldsymbol{O}), \boldsymbol{H} \rangle \quad \text{for all } \boldsymbol{H} \in \mathbb{V}. \tag{A.2}$$

The *Riemannian gradient* $\boldsymbol{G}(\boldsymbol{O})$, which serves as the (negative) search direction for a gradient descent method on manifolds, is simply the orthogonal projection of $\nabla f(\boldsymbol{O})$ onto the tangent space $T_{\boldsymbol{O}}\mathcal{M}$, i.e.

$$\boldsymbol{G}(\boldsymbol{O}) = \Pi_{T_{\boldsymbol{O}}\mathcal{M}}\big(\nabla f(\boldsymbol{O})\big), \tag{A.3}$$

with $\Pi_{T_{\boldsymbol{O}}\mathcal{M}}$ denoting the orthogonal projection with respect to $\langle \cdot, \cdot \rangle$. Now let $t \mapsto \Gamma(\boldsymbol{O}, \boldsymbol{H}, t)$ denote the geodesic emanating from $\boldsymbol{O} \in \mathcal{M}$ in direction $\boldsymbol{H} \in T_{\boldsymbol{O}}\mathcal{M}$, that is

$$\Gamma(\boldsymbol{O}, \boldsymbol{H}, 0) = \boldsymbol{O} \quad \text{and} \quad \tfrac{\mathrm{d}}{\mathrm{d}t}\big|_{t=0} \Gamma(\boldsymbol{O}, \boldsymbol{H}, t) = \boldsymbol{H}. \tag{A.4}$$

Schematically, line search methods on manifolds update the i-th estimate $\boldsymbol{O}^i$ by a step along the curve

$$\boldsymbol{O}^{i+1} = \Gamma(\boldsymbol{O}^i, \boldsymbol{H}^i, t^i), \tag{A.5}$$

where $\boldsymbol{H}^i \in T_{\boldsymbol{O}^i}\mathcal{M}$ is the descent direction and $t^i \in \mathbb{R}$ is a suitable step-size.

In practice, faster convergence can often be achieved by adapting conjugate gradient methods to the manifold setting. In this case, the search direction $\boldsymbol{H}^{i+1} \in T_{\boldsymbol{O}^{i+1}}\mathcal{M}$ is a linear combination of the Riemannian gradient $\boldsymbol{G}^{i+1} := \boldsymbol{G}(\boldsymbol{O}^{i+1}) \in T_{\boldsymbol{O}^{i+1}}\mathcal{M}$ and the previous search direction $\boldsymbol{H}^i$. Since linear combinations of elements from different tangent spaces are not defined, parallel transport along geodesics is used to associate the different tangent

spaces. Let this parallel transport be denoted by

$$\Psi_i^{i+1} \colon T_{\boldsymbol{O}^i}\mathcal{M} \to T_{\boldsymbol{O}^{i+1}}\mathcal{M}, \tag{A.6}$$

the conjugate gradient method on manifold updates the search direction via

$$\boldsymbol{H}^{i+1} := -\boldsymbol{G}^{i+1} + \beta^i \, \Psi_i^{i+1}(\boldsymbol{H}^i), \tag{A.7}$$

where initially, $\boldsymbol{H}^0 := -\boldsymbol{G}^0$. For the implementation used in this thesis, the update parameter β^i is chosen according to a manifold adaption of the Fletcher-Reeves and Dai-Yuan formula. More precisely, a hybridization of the Hestenes-Stiefel and the Dai-Yuan formula is employed

$$\beta_{hyb}^i = \max\big(0, \min(\beta_{DY}^i, \beta_{HS}^i)\big), \tag{A.8}$$

which was suggested in [44], where

$$\beta_{HS}^i = \frac{\langle \boldsymbol{G}^{i+1}, \boldsymbol{G}^{i+1} - \Psi_i^{i+1}(\boldsymbol{G}^i)\rangle}{\langle \Psi_i^{i+1}(\boldsymbol{H}^i), \boldsymbol{G}^{i+1} - \Psi_i^{i+1}(\boldsymbol{G}^i)\rangle}, \tag{A.9}$$

$$\beta_{DY}^i = \frac{\langle \boldsymbol{G}^{i+1}, \boldsymbol{G}^{i+1}\rangle}{\langle \Psi_i^{i+1}(\boldsymbol{H}^i), \boldsymbol{G}^{i+1} - \Psi_i^{i+1}(\boldsymbol{G}^i)\rangle}. \tag{A.10}$$

Using this new search direction, the new iterate $\boldsymbol{O}^{i+1}$ is obtained through Eq. (A.5) by moving along the geodesic emanating from $\boldsymbol{O}^i$ in the search direction with a step size t^i. Generally, the ideal step size is found by solving

$$\hat{t}^i := \underset{t^i>0}{\arg\min} f(\Gamma(\boldsymbol{O}^i, \boldsymbol{H}^i, t^i)). \tag{A.11}$$

To avoid having to solve this sub-problem in every iteration, one can perform an *Armijo line-search* instead. This involves setting a large initial step size t_0^i and incrementally decreasing it by a constant factor $0 < c_1 < 1$ until the Armijo condition

$$f(\Gamma(\boldsymbol{O}^i, \boldsymbol{H}^i, t^i)) \leq f(\boldsymbol{O}^i) + c_2 t^i \langle \boldsymbol{G}^i, \boldsymbol{H}^i\rangle \tag{A.12}$$

is met [128], typically with very small $0 < c_2 < 1$.

Appendix B: Derivation of gradients

B.1 Derivation of the Riemannian gradient in Section 4.5

In this section, the Riemannian gradient of Eq. (4.25) that is required for the bimodal alignment algorithm is derived. Let $\langle \cdot, \cdot \rangle_{\boldsymbol{P}}$ be the Riemannian metric on the Lie group $\mathcal{G}$ inherited from Eq. (4.24) and let $F(\cdot)$ be a smooth real valued function on $\mathcal{G}$. Then the Riemannian gradient of F at $\delta \in \mathcal{G}$ is the unique vector $\boldsymbol{G} \in T_\delta \mathcal{G}$, with $T_\delta \mathcal{G}$ as the tangent space at δ, such that

$$\frac{\mathrm{d}}{\mathrm{d}t}\Big|_{t=0} F(e^{t\boldsymbol{H}}\delta) = \langle \boldsymbol{H}, \boldsymbol{G} \rangle_{\boldsymbol{P}} \tag{B.1}$$

holds for all tangent elements $\boldsymbol{H} \in T_\delta \mathcal{G}$.

For the purpose at hand, the gradient is computed at $\delta = \mathrm{id}$. Now let B be the image region in which the modalities I_U and I_V should be aligned. One assumes that B is rectangular and

$$I(\boldsymbol{x})_{\boldsymbol{x} \in B} \tag{B.2}$$

denotes the vectorized version of I over the domain B. Using Eq. (4.23) and the fact that $\boldsymbol{c} := \boldsymbol{\Omega}_U^F I_U$ is a constant vector,

$$\begin{aligned} \tfrac{\mathrm{d}}{\mathrm{d}t}\Big|_{t=0} F(e^{t\boldsymbol{H}}\tau) &= \tfrac{\mathrm{d}}{\mathrm{d}t}\Big|_{t=0} g\left(\boldsymbol{c}, \boldsymbol{\Omega}_V^F\left[(e^{t\boldsymbol{H}}\tau) \circ I_V\right]\right) \\ &= \nabla g\left(\boldsymbol{c}, \boldsymbol{\Omega}_V^F I_V(\tau\boldsymbol{x})_{\boldsymbol{x}\in B}\right)^\top \boldsymbol{\Omega}_V^F \left[\tfrac{\mathrm{d}}{\mathrm{d}t}\Big|_{t=0} I_V(e^{t\boldsymbol{H}}\tau\boldsymbol{x})_{\boldsymbol{x}\in B}\right] \end{aligned} \tag{B.3}$$

holds by applying the chain rule. The last bracket is a vector where each of its entries is

computed as

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\Big|_{t=0} I_V(e^{t\boldsymbol{H}}\tau\boldsymbol{x}) &= \nabla I_V(\tau\boldsymbol{x})^\top \boldsymbol{H}\tau\boldsymbol{x} \\ &= \mathrm{vec}(\tau\boldsymbol{x}\otimes\nabla I_V(\tau\boldsymbol{x}))^\top \mathrm{vec}(\boldsymbol{H}),\end{aligned} \tag{B.4}$$

where, $\mathrm{vec}(\cdot)$ denotes the linear operator that stacks the columns of a matrix among each other and $\otimes$ is the Kronecker product. Note that since the representation with homogeneous coordinates is used, $\nabla I_V(\boldsymbol{x}) \in \mathbb{R}^3$ is the common image gradient of I_V with an additional 0 in the third component.

Thus, with

$$\boldsymbol{r}^\top := \nabla g\left(\boldsymbol{c}, \boldsymbol{\Omega}_V^F I_V(\tau\boldsymbol{x})_{\boldsymbol{x}\in B}\right)^\top \boldsymbol{\Omega}_V^F \left(\mathrm{vec}(\tau\boldsymbol{x}\otimes\nabla I_V(\tau\boldsymbol{x}))^\top\right)_{\boldsymbol{x}\in B}, \tag{B.5}$$

one obtains

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\Big|_{t=0} F(e^{t\boldsymbol{H}}\delta) &= \boldsymbol{r}^\top \mathrm{vec}(\boldsymbol{H}) \\ &= \mathrm{tr}(\mathrm{vec}^{-1}(\boldsymbol{r})\boldsymbol{H}^\top) \\ &= \langle \mathrm{vec}^{-1}(\boldsymbol{r}) \odot \hat{\boldsymbol{P}}, \boldsymbol{H}\rangle_{\boldsymbol{P}},\end{aligned} \tag{B.6}$$

where the entries of $\hat{\boldsymbol{P}}$ are the inverse of the entries of $\boldsymbol{P}$.

Using Eq. (B.1), the Riemannian gradient is the orthogonal projection of $\mathrm{vec}^{-1}(\boldsymbol{r}) \odot \hat{\boldsymbol{P}}$ with respect to $\langle\cdot,\cdot\rangle_{\boldsymbol{P}}$ onto the tangent space of $\delta = \mathrm{id}$, which is the Lie algebra $\mathfrak{g}$, i.e.

$$\mathrm{grad}_\delta F(\delta\circ\tau) = \Pi_{\mathfrak{g}}\left(\mathrm{vec}^{-1}(\boldsymbol{r}) \odot \hat{\boldsymbol{P}}\right). \tag{B.7}$$

If for the entries p_{ij} of $\boldsymbol{P}$ it is further assumed that

$$p_{11} = p_{22} \text{ and } p_{12} = p_{21}, \tag{B.8}$$

then, for the considered Lie groups, these projections are explicitly given by

$$\Pi_{SO}(\boldsymbol{X}) = \begin{bmatrix} \frac{1}{2}(\boldsymbol{X}_{11} - \boldsymbol{X}_{11}^\top) & 0 \\ 0 & 0 \end{bmatrix} \tag{B.9}$$

$$\Pi_{SE}(\boldsymbol{X}) = \begin{bmatrix} \frac{1}{2}(\boldsymbol{X}_{11} - \boldsymbol{X}_{11}^\top) & \boldsymbol{x}_{12} \\ 0 & 0 \end{bmatrix} \tag{B.10}$$

$$\Pi_{SA}(\boldsymbol{X}) = \begin{bmatrix} (\boldsymbol{X}_{11} - \frac{1}{2}\operatorname{tr}(\boldsymbol{X}_{11})\boldsymbol{I}_2) & \boldsymbol{x}_{12} \\ 0 & 0 \end{bmatrix} \tag{B.11}$$

$$\Pi_{A}(\boldsymbol{X}) = \begin{bmatrix} \boldsymbol{X}_{11} & \boldsymbol{x}_{12} \\ 0 & 0 \end{bmatrix}, \tag{B.12}$$

where $\boldsymbol{X} \in \mathbb{R}^{3\times 3}$ is partitioned as

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_{11} & \boldsymbol{x}_{12} \\ \boldsymbol{x}_{21}^\top & x_{22} \end{bmatrix}. \tag{B.13}$$

B.2 Derivation of the Euclidean gradient in Section 5.3

The Euclidean gradient required in the numerical optimization of the filter learning problem described in (5.17) is derived as follows. The cost function to minimize in the learning stage consists of three terms, one each for the approximated cost of the jump set $f(\boldsymbol{\Phi})$, the centroid penalty $r(\boldsymbol{\Phi})$ and the coherence penalty $h(\boldsymbol{\Phi})$.

Sparsity objective: First, the derivative of the approximated cost of the jump set is provided. Explicitly, let $\boldsymbol{\Phi}_k \in \mathbb{R}^n$ represent a vectorized 2D filter of size $\sqrt{n} \times \sqrt{n}$ that is applied to a set of M vectorized 2D image patches $\boldsymbol{U} \in \mathbb{R}^{n\times M}$ by taking their standard inner product $\boldsymbol{\Phi}_k^\top \boldsymbol{U} \in \mathbb{R}^{1\times M}$. For a set of filters, one obtains $\boldsymbol{\Phi U} \in \mathbb{R}^{K\times M}$ accordingly. By denoting $\mathbf{D}_{a_s} = \nabla_{a_s}\sigma(\boldsymbol{\Phi}\mathbf{U})$ shorthand for the difference of features along $\boldsymbol{a}_s$ and using $\odot$

for the Hadamard product, one obtains

$$\frac{\partial}{\partial \boldsymbol{\Phi}} f(\boldsymbol{\Phi}) = 4\nu\mu \sum_{s=1} \omega_{a_s} \Bigg[\left(\frac{\mathbf{D}_{a_s}}{1+\nu\|\mathbf{D}_{a_s}\|^2} \odot \frac{\boldsymbol{\Phi}\mathbf{U}_{a_s}}{1+\mu(\boldsymbol{\Phi}\mathbf{U}_{a_s})^2} \right) \mathbf{U}_{a_s}^\top - \left(\frac{\mathbf{D}_{a_s}}{1+\nu\|\mathbf{D}_{a_s}\|^2} \odot \frac{\boldsymbol{\Phi}\mathbf{U}_0}{1+\mu(\boldsymbol{\Phi}\mathbf{U}_0)^2} \right) \mathbf{U}_0^\top \Bigg] \quad \text{(B.14)}$$

for the derivative of f. Here, $\mathbf{U}_{a_s}$ is the $n \times M$ data matrix containing vectorized patches $\mathbf{U}_i^{a_s}$ cropped from the M sampled super-patches in direction $\boldsymbol{a}_s$ according to Figure 5.2.

Centroid penalty: Second, the derivative of the centroid constraint in Eq. (5.16) is required. This constraint acts on each filter independently. For the individual filter, one gets

$$\frac{\partial}{\partial \boldsymbol{\Phi}_k} h(\boldsymbol{\Phi}) = \frac{4}{w_-} \left[\frac{c_{k,x}\boldsymbol{P}_x}{1-c_{k,x}^2} + \frac{c_{k,y}\boldsymbol{P}_y}{1-c_{k,y}^2} + \frac{1}{2}(c_{k,x}-c_{k,y})(\boldsymbol{P}_x - \boldsymbol{P}_y) \right] \boldsymbol{\Phi}_k \quad \text{(B.15)}$$

by using $w_- = \frac{\sqrt{n}-1}{2}$ as a shorthand notation for the half width of the filter. By stacking the individual derivatives, the derivative of h with respect to the entire filter set can then be written as

$$\frac{\partial}{\partial \boldsymbol{\Phi}} h(\boldsymbol{\Phi}) = \left[\frac{\partial}{\partial \boldsymbol{\Phi}_1} h(\boldsymbol{\Phi}), \ldots, \frac{\partial}{\partial \boldsymbol{\Phi}_K} h(\boldsymbol{\Phi}) \right]^\top. \quad \text{(B.16)}$$

Coherence penalty: Last, the gradient of the coherence penalty in Eq. (5.12) is provided in [74] as

$$\frac{\partial}{\partial \boldsymbol{\Phi}} r(\boldsymbol{\Phi}) = \left[\sum_{1 \le i < j \le k} \frac{2\boldsymbol{\Phi}_i^\top \boldsymbol{\Phi}_j}{1-\left(\boldsymbol{\Phi}_i^\top \boldsymbol{\Phi}_j\right)^2} \left(\mathcal{E}_{ij} + \mathcal{E}_{ji}\right) \right] \boldsymbol{\Phi}. \quad \text{(B.17)}$$

Here $\mathcal{E}_{ij}$ is a matrix with a one in component ij and zero elsewhere.

Finally, the gradient of the cost function in Eq. (5.17) is obtained by combining Eq. (B.14), Eq. (B.16) and Eq. (B.17), yielding

$$\nabla E(\boldsymbol{\Phi}) = \frac{\partial}{\partial \boldsymbol{\Phi}} f(\boldsymbol{\Phi}) + \lambda \frac{\partial}{\partial \boldsymbol{\Phi}} r(\boldsymbol{\Phi}) + \kappa \frac{\partial}{\partial \boldsymbol{\Phi}} h(\boldsymbol{\Phi}). \quad \text{(B.18)}$$