

**Learning Low-Dimensional Representations
With Application
to Classification and Decision-Making**

Vom Fachbereich 18
Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt
zur Erlangung der Würde eines
Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Dissertation

von
Jürgen Hahn, M.Sc.
geboren am 29. Dezember 1985 in Berlin-Friedrichshain

Referent:	Prof. Dr.-Ing. Abdelhak M. Zoubir
Korreferent:	Prof. Sergios Theodoridis, Ph.D.
Tag der Einreichung:	13. September 2016
Tag der mündlichen Prüfung:	16. Dezember 2016

Acknowledgments

I wish to thank all the people who helped and supported me in the recent years.

First of all, I would like to thank Prof. Dr.-Ing. Zoubir for giving me the opportunity to work in an excellent research group. I am grateful for his support and especially for giving me the chances to collect international experience during international conferences and my stay as visiting researcher at the University of Wollongong. Further, I would like to thank Prof. Sergios Theodoridis, Ph.D., for his co-supervision.

Moreover, I wish to thank Prof. Bouzerdoum, Ph.D., head of the Visual and Audio Signal Processing Lab at the University of Wollongong, and his group for the hospitality and the fruitful discussions. Without his outstanding support, it would not have been possible to organize a research stay in such a short time.

I am very grateful for the opportunity to work with so many great people during my five years as PhD candidate. First of all, I want to thank my room mates, Christian Weiß, Adrian Šošić, and Simon Rosenkranz for all the (sometimes pointless) discussions we had. I also enjoyed working with Sergey Sukhanov, Michael Lang, Roy Howard, Dominik Reinhard, Ann-Kathrin Seifert, Patrica Binder, Freweyni Teklehaymanot, Sahar Khawatmi, Khadidja Hamaidi, Dr.-Ing. Michael Fauss, Dr.-Ing. Michael Muma, Mark Balthasar, Di Jin, Tim Schäck, Dr.-Ing. Nevine Demitri, Dr.-Ing. Sara Al-Sayed, Dr.-Ing. Feng Yin, and Wassim Suleiman. I appreciate your help, comments, and ideas. Further, I want to thank the former SPG members who guided me well in the beginning of my PhD: Dr.-Ing. Philip Heidenreich, Dr.-Ing. Stefan Leier, Dr.-Ing. Mouhammad Alhumaidi, and Dr.-Ing. Raquel Fandos.

I especially want to thank Renate Koschella und Hauke Fath for taking care of any administrative and technical problem.

I also want to thank the people I had the opportunity to collaborate with, Dr.-Ing. Marco Moebus (Adam Opel AG) and Dr.-Ing. Christian Debes (AGT International). Especially I wish to thank the people from AGT for letting me be part of their team for the data fusion contest: Roel Heremans, Ph.D., Andreas Merentitis, Ph.D., Nikolaos Frangiadakis, Ph.D., Tim van Kasteren, Ph.D., and Dr.-Ing. Christian Debes.

During my doctoral studies I supervised excellent students whose work contributed to this thesis. Therefore, I want to thank Vineet Kumar, Platini Rodrigue Ngankam, Jiao Du, Alexander Zorn, Lisa Hesse, and Patrick Wenzel.

Further, I wish to thank Yvonne Späck-Leigsnering, Jens Thekkeveettil, Dr.-Ing. Jean Kadavelil, Markus Hessinger, and Carolin Reimann.

Finally, I wish to express my sincere thanks to my family: my sister Viviane and her husband Hüsamettin, my lovely nieces Helin and Nisa, and my parents Esther and Eberhard. Without your unconditional love, patience, and trust, this work would not have been possible.

Darmstadt, January 28th, 2017

Kurzfassung

Viele Algorithmen in der Signalverarbeitung und des Maschinellen Lernens erzielen bei der Anwendung auf hochdimensionalen Daten suboptimale Ergebnisse, wie durch das Phänomen des *Fluchs der Dimensionen* bekannt ist. Das Erlernen niedrigdimensionaler Repräsentationen zielt darauf ab, die Dimensionalität des Beobachtungsraumes zu reduzieren, wobei charakteristische Merkmale der Daten erhalten werden sollen. Darüberhinaus können diese Darstellungen helfen, Strukturen in den Daten zu entdecken, die tiefere Erkenntnisse über die Beobachtungen ermöglichen. Aus diesen Gründen werden in dieser Arbeit Modelle zum Erlernen niedrigdimensionaler Repräsentationen vorgeschlagen, die eine Analyse der beobachteten Daten ermöglichen. Insbesondere werden Ansätze zur effizienten Aufzeichnung, Klassifikation sowie zum Erlernen der Strukturen in den beobachteten Daten präsentiert.

Zunächst werden niedrigdimensionale Methoden für die Klassifikation mit Anwendung in der hyperspektralen Bildgebung vorgestellt. In der Fernerkundung stellt die hyperspektrale Bildgebung ein effizientes Mittel zur Analyse großer Areale bereit. Insbesondere können mit den aufgezeichneten Daten verschiedene Materialien einfach unterschieden werden, da jedes Element des aufgezeichneten Bildes das Spektrum des sichtbaren sowie des infraroten Lichts darstellt. Für die Klassifikation wird ein Ansatz zur Auswahl der Merkmale und zur effizienten Aufzeichnung der Daten vorgestellt. Das Ziel beider Methoden besteht darin, die Menge der Daten, die während der Klassifikation ausgewertet werden muss, zu reduzieren, wobei hohe Klassifikationsgenauigkeiten erhalten werden sollen. Im ersten Ansatz wird eine *cluster*-basierte Methode zur Auswahl der Bänder des hyperspektralen Bildes vorgestellt, welche als Merkmale für die Klassifikation betrachtet werden können. Da das Entfernen aufwändig aufgezeichneter Daten zu einer ineffizienten Nutzung der Ressourcen führt, wird weiterhin ein Ansatz zur Aufzeichnung basierend auf *Compressive Sensing* vorgestellt. Der Kerngedanke dieser Methode besteht darin, die Daten in einer niedrigdimensionalen Darstellung aufzuzeichnen. Für die Klassifikation können diese Daten als eingebettet in einem Merkmalraum betrachtet werden. Da wir direkt an einer Klassifikation interessiert sind, ist eine aufwendige Rekonstruktion der Daten nicht notwendig und kann vermieden werden.

Weiterhin wird ein merkmalsbasiertes Verfahren zum Erlernen von Strukturen in den Spektren vorgeschlagen, welches die in den hyperspektralen Bildern vorkommenden Materialien extrahiert. Hyperspektrale Bilder weisen oft eine geringe räumliche Auflösung auf, so dass die Elemente des Bildes große Areale darstellen, oft im Bereich von 2 m^2 bis 400 m^2 . Viele Algorithmen, die u.a. zur Klassifikation verwendet werden, nehmen jedoch an, dass jedes Bildelement nur ein einzelnes Material darstellt. Folglich

ist das Erlernen der Struktur eine wichtige Aufgabe in der hyperspektralen Bildgebung. Dies wird auch als spektrales *unmixing* bezeichnet. Hierfür wird eine Bayessche nicht-parametrische Formulierung des Problems vorgeschlagen. Ein wichtiger Vorteil dieses Modells im Vergleich zu bestehenden Methoden ist, dass die Anzahl der Materialien aus den Daten inferiert werden kann und somit nicht als bekannt vorausgesetzt wird. Die vorgeschlagene Formulierung resultiert in einem Bayesschen nichtparametrischen *unmixing* Algorithmus, der in der Lage ist, die Anzahl der Merkmale, die Merkmale selbst, sowie deren Koeffizienten gemeinsam zu lernen.

Zuletzt wird ein Modell für die Entscheidungsfindung vorgeschlagen, das auf Merkmalsdarstellungen der Beobachtungen basiert. Insbesondere wird das Problem des Lernens aus Beobachtungen mit dem Ziel betrachtet, ein Verhalten aus den Beobachtungen eines erfahrenen Agenten zu lernen. Ein bedeutender Unterschied zu bestehenden Verfahren liegt in der Annahme, dass der Agent seine Entscheidungen basierend auf Merkmalen der Beobachtungen trifft, wobei jedes Merkmal eine bestimmte Entscheidung impliziert. Das Erlernen der Merkmale und deren Verhaltensregeln ermöglicht es Schlüsse über das beobachtete Verhalten zu ziehen. Weiterhin können Aktionen für neue Situationen vorhergesagt werden, aus denen Verhaltensregeln für andere Agenten abgeleitet werden können. Um die Möglichkeiten des Modells anhand eines realen Problems aufzuzeigen, wird mithilfe des entwickelten Algorithmus das Verhalten eines Fahrers analysiert, welches eine typische Aufgabe in intelligenten Fahrerassistenzsystemen ist.

Die Algorithmen, die auf den vorgeschlagenen Modellen basieren, werden mithilfe simulierter Daten zur Überprüfung der Konzepte ausgewertet. Weiterhin werden alle Methoden auf reale Daten angewandt, um die Leistungsfähigkeit der entwickelten Ansätze zu demonstrieren.

Abstract

Many signal processing and machine learning algorithms perform poorly when applied to high-dimensional data, as is known by the phenomenon of the *curse of dimensionality*. Learning low-dimensional representations aims at reducing the dimensionality of the observation space while maintaining the characteristics of the data. Further, low-dimensional representations can help to reveal latent structures, allowing for deeper insights into the observations. For these reasons, models are proposed that allow to learn low-dimensional representations of the observations, providing means for the analysis of the observed data. In particular, approaches for efficient data acquisition and classification and for the inference of the structure of the observed data are presented.

First, low-dimensional methods for classification are proposed with application to hyperspectral imaging. In remote sensing, hyperspectral imaging provides an efficient means for the analysis of vast areas. As each element of the captured image represents the spectrum of the visible and infra-red light, the acquired data allows for effective discrimination between different materials. For classification, a feature selection approach as well as a sparse acquisition scheme are presented. The goal of both methods is to reduce the amount of data that needs to be evaluated during classification, while maintaining high classification accuracies. In the first approach, a clustering-based method for selecting the bands of a hyperspectral image, which can be considered as features for classification, is proposed. However, removing costly acquired data during feature selection is clearly resource-inefficient. For this reason, further a sparse acquisition approach based on the Compressive Sensing framework is proposed. The key idea of this approach is to capture the data in a low-dimensional representation, which is interpreted as being embedded in a feature space for the classification problem. As we are interested in the classification result directly, costly reconstruction of the data is not required and can be avoided.

Second, a feature-based approach to learn the structure of the spectra is proposed, revealing the materials present in a hyperspectral image. Hyperspectral images often suffer from low spatial resolutions such that each element of an image represents a large area, often in the range from 2 m² to 400 m². However, many algorithms, such as for classification, assume that each element of the image represents a single material only. Thus, learning the structure is an important task in the analysis of hyperspectral images, which is also known as spectral unmixing. For this, a Bayesian nonparametric formulation of the problem is proposed. A significant advantage of this model, in comparison to existing approaches, is that the number of materials is inferred from the data and, hence, is not required to be known *a priori*. The proposed formulation

results in a Bayesian nonparametric unmixing algorithm which enables to learn the number of latent features, the actual features, and their coefficients jointly.

Third, a model for decision-making based on a feature representation of the observations is proposed. In particular, the problem of learning from observations is considered, in which we aim at learning a behavior from observations which are provided by an experienced agent. A key difference to existing approaches consists in the assumption that the agent makes its decision based on latent features of the observations, where each feature indicates a certain action. Learning the features and their policies enables to reason about the observed behavior. Further, actions for new situations can be predicted, from which a policy can be derived for other agents. Using the developed algorithm, a driver's behavior is analyzed, which is a typical task in advanced driver assistance systems, in order to show the performance of the model in a real-world problem.

The algorithms based on the proposed models are evaluated on simulated data to proof the concepts. Further, all methods are applied to real data, demonstrating the high performance of the developed approaches.

Contents

1	Introduction and Motivation	1
1.1	State of the Art	2
1.2	Contributions	5
1.3	Publications	6
1.4	Organization of the Thesis	7
2	Machine and Feature Learning Fundamentals	9
2.1	Computational Learning Theory	9
2.1.1	Statistical Learning Theory	9
2.1.2	Bayesian Inference	11
2.1.3	Curse of Dimensionality	12
2.2	Approximate Inference Techniques	13
2.2.1	Metropolis and Metropolis-Hastings Sampling	13
2.2.2	Gibbs Sampling in a Directed Graphical Model	14
2.2.3	Parallel Tempering	16
2.3	Feature Learning and Low-Dimensional Representations	17
2.3.1	Feature Models	17
2.3.2	Low-Dimensional Data Acquisition Using Compressive Sensing	18
2.3.3	Bayesian Feature Learning	19
2.4	Clustering and Classification	24
2.4.1	Spectral Clustering	24
2.4.2	k -Nearest Neighbor	26
2.4.3	Support Vector Machine	27
3	Feature Learning for Classification in Hyperspectral Imaging	33
3.1	Motivation and Introduction	33
3.2	Hyperspectral Image Data Sets	35
3.2.1	Indiana’s Indian Pines (IP)	35
3.2.2	University of Pavia (UP) and Center of Pavia (CP)	36
3.2.3	Acknowledgment	38
3.3	Band Selection by Means of Spectral Clustering	38
3.3.1	Cluster Formation	38
3.3.2	Representative Selection	39
3.3.3	Classification	40
3.3.4	Results	41
3.4	Classification in the Compressive Domain	44
3.4.1	Compressed Classification	45

3.4.2	System Design	46
3.4.3	Optimization of the Acquisition Process	47
3.4.4	Results	51
3.5	Conclusions	53
4	Hyperspectral Unmixing via Bayesian Nonparametric Feature Learning	55
4.1	Motivation and Introduction	55
4.2	State of the Art	56
4.3	Contribution	57
4.4	Bayesian Nonparametric Unmixing Model	58
4.4.1	Observation Likelihood	59
4.4.2	Prior for the Noise Variance	59
4.4.3	Prior for the Abundances	60
4.4.4	Prior for the Endmember Weights and Activations	60
4.4.5	Joint Posterior Distribution	61
4.5	Inference	61
4.5.1	Sampling the Noise Variance	61
4.5.2	Sampling the Abundances	62
4.5.3	Sampling the Endmember Weights	62
4.5.4	Sampling the Endmember Activations	63
4.5.5	Sampling Procedure	65
4.6	Experimental Results	66
4.6.1	Simulations	69
4.6.2	Real Data	76
4.7	Discussion and Outlook	80
4.8	Conclusion	82
5	Bayesian Feature-Based Learning From Demonstrations	83
5.1	Motivation and Introduction	83
5.2	Contributions	84
5.3	Problem Formulation	85
5.4	State of the Art	86
5.4.1	Reinforcement Learning	86
5.4.2	Inverse Reinforcement Learning	87
5.4.3	Imitation Learning	88
5.5	Choice of the Model	88
5.5.1	Feature Model for Learning From Demonstrations	88
5.5.2	Transition Model	89
5.5.3	Feature-Based Policy	90

5.5.4	Alternative Feature-Based Models	90
5.6	Bayesian Nonparametric Model for Feature Learning	92
5.6.1	Observation Likelihood and Noise Variance	92
5.6.2	Prior for the Feature Weights and Activations	92
5.6.3	Prior for the Substates	93
5.6.4	Mixture of Policies and Action Likelihood	94
5.6.5	Joint Posterior Distribution	95
5.7	Inference	95
5.7.1	Conditional Distributions	96
5.7.2	Sampling Algorithm	99
5.7.3	Prediction of Actions	100
5.8	Experimental Results	101
5.8.1	Estimation of the Features	102
5.8.2	Prediction of Actions	102
5.8.3	Estimation of the Number of Features	104
5.9	Real Data Experiments	106
5.9.1	Scene 11 - Traffic Jam	108
5.9.2	Scene 20 - Lane Change	113
5.10	Discussion	114
5.11	Conclusion	116
6	Conclusions and Outlook	117
6.1	Summary and Conclusions	117
6.1.1	Feature Learning for Classification in Hyperspectral Imaging . .	117
6.1.2	Hyperspectral Unmixing via Bayesian Nonparametric Feature Learning	118
6.1.3	Feature-Based Decision-making	119
6.2	Outlook	119
6.2.1	Low-Dimensional Representations for Classification in Hyperspectral Imaging	119
6.2.2	Features for the Analysis of Hyperspectral Images	120
6.2.3	Features for the Analysis and Prediction in Decision-Making Problems	121
A	Derivations for Compressive Classification	123
B	Derivations for Bayesian Nonparametric Feature Learning	125
B.1	Conditionals for the Feature Weights	125
B.1.1	Using the Distance Prior	126
B.1.2	Using an Exponential Prior	127

B.1.3 Using a Gaussian Prior	128
B.2 Conditional for the Noise Variance	129
B.3 Sparsity-Promoting Prior on the Feature Coefficients	129
List of Acronyms	131
List of Symbols	135
Bibliography	145
Curriculum Vitae	161

Chapter 1

Introduction and Motivation

Processing large amounts of data has become increasingly important in the recent past and is required in a wide range of applications. Prominent examples can be found in remote sensing [1], social networks [2], recommender systems [3], and intelligent robotics [4]. These applications usually incorporate regression, classification, and decision-making where many data samples (observations) need to be evaluated such that conclusions can be drawn. A key problem of many algorithms used for data analysis is their poor performance in case of high-dimensional input data – a phenomenon which is widely known as the curse of dimensionality [5].

Often, a feature representation is chosen to reduce the dimensionality of the data and to overcome the problems induced by the curse of dimensionality. For this purpose, it is typically assumed that the relevant information contained in the observations is embedded in a subspace. Thus, features enable for a problem-specific representation of the data in which important characteristics are retained, while irrelevant information is reduced. A crucial question for many data analysis methods, e.g., for classification and regression, is the choice of features. Depending on the feature representation, the analysis may succeed or, in the worst case, fail completely. In many applications, features are handcrafted, requiring an expert of the respective domain to design them. Often, the data that shall be analyzed is complex and the design of the features is carried out in a trial-and-error approach. Thus, it is desirable to learn features from the observed data directly [6].

Besides reducing the size of the data to increase the efficiency of the post-processing steps, such as classification, low-dimensional representations can also be used to learn the structure of the data. In many applications, features have a specific meaning and can, therefore, be interpreted by experts. If we are able to learn the features present in the data in compliance with our prior assumptions, we can obtain a deeper understanding of the observations. An example can be found in hyperspectral imaging. Here, the features of the pixels can be considered as the spectral signatures of the pure materials present in the captured scene [7, 8].

In this thesis, we address the problem of learning low-dimensional representations from observed data. We refer to this problem as Feature Learning (FL). As the transformation into the low-dimensional space as well as the projections of the observations are

unknown, this problem can be considered as a Blind Source Separation (BSS) problem. Many methods have been developed in the past to solve BSS problems, famous examples are Principal Component Analysis (PCA) [9, 10], Independent Component Analysis (ICA) [11], and Non-negative Matrix Factorization (NMF) [12]. However, most methods make specific assumptions about the observed data and their structure that may not hold for the problem at hand. Further, many algorithms assume the number of latent features to be known, which is rarely the case in practice.

In this thesis, we propose different feature selection and learning techniques that aim at improving post-processing steps and allow for an accurate analysis of the data. In particular, we investigate statistical methods for low-dimensional data acquisition and Bayesian models to identify the latent structure of the data. For the latter, we provide means to infer the number of features by making use of Bayesian Nonparametric techniques [13]. At this point we would like to clarify, though FL is generally not restricted to learning low-dimensional representations, e.g., kernel methods often operate on a feature space larger than the space of observations, we understand FL as learning low-dimensional representations of the observations. Throughout the thesis, we thus use the terms *low-dimensional representations* and *features representations* interchangeably.

1.1 State of the Art

Various feature selection and learning methods have been proposed in the past. An exhaustive overview about feature learning would fill a book on its own. Thus, in this section, we concentrate on important work that is relevant for this thesis.

While in feature selection, one assumes that a set of features is available, where the task is to estimate the presence of the features in the observations, FL aims at estimating the features itself. As explained in [14], one can distinguish between two categories of FL: (i) extrinsic and (ii) intrinsic methods. Extrinsic methods are independent of the post-processing algorithm and, therefore, can be used with many methods. For example, PCA [9, 10], ICA [11], and NMF [12] belong to this category. Due to their nature, application-specific constraints can often not be incorporated and assumptions are being made that may not hold for the problem at hand. In contrast, intrinsic methods are integrated in the post-processing algorithm, and thus need to be designed or at least be adapted to the algorithm and application. A general framework for this purpose is Bayesian Feature Learning (BFL) [15–17] which is formulated as a generative model and, thus, can be adapted and extended to the problem.

In some research communities, features are also referred to as entries of a codebook or dictionary. One of the most prominent and successful approaches to FL is sparse coding [18], where a linear latent feature model is derived with sparse feature coefficients. Further, a dictionary is learned by iterating between the estimation of the coefficients and the dictionary. Sparse coding has shown good results in computer vision [19–21] and audio applications [22]. There have been several extensions to this work. For example, in [23], Local Coordinate Coding (LCC) is presented that exploits local linear models, with the goal of fast learning large dictionaries. A high dimensional dictionary is learned by k -means clustering which is used to transform the observations into a sparse representation. Locality is exploited by assuming a linear relation between the observations and the basis spanned by the k -nearest neighbors.

In the recent past, spike and slab models have demonstrated excellent performance for feature learning. Spike and slab models assume that a variable is composed of the latent spike, a binary variable, and the latent slab, a variable of a domain defined by the application [24, 25]. In feature learning, the slab corresponds to the weight of an element of a feature vector. The spike can be understood as a feature activation that activates the corresponding weight depending on the observed data samples [26]. The concept of sparse coding has been combined with the spike and slab model into a deep architecture in [27], showing high accuracies in image classification as well as transfer learning.

FL has changed significantly since the development of deep architectures in the context of the Deep Learning (DL) framework. The idea of stacking multiple processing layers to learn features and perform classification, e.g., using several hidden layers in a neural network, posed huge challenges for training in the past. Naive gradient-based approaches (such as the backpropagation algorithm [28, 29]) using random initialization of the network weights usually fail at learning networks with several hidden layers. There are two potential explanations for this. First, the noise in the gradient increases with each layer, providing insufficient information about the parameters to be learned [30]. Second, due to the high flexibility of the model, overfitting is likely to occur and effective regularization is needed [30]. In 2006, a new method for training multi-layered networks has been proposed, referred to as *pre-training*, which initializes the layers by means of unsupervised training [6, 30–32]. Since pre-training does not require class information, which is usually expensive to obtain, the network can easily be trained on huge amounts of data. Given a good initialization of the network layers, a refinement of the parameters is conducted using conventional supervised learning techniques. As the hidden layers can be understood as feature hierarchies [32–34], DL provides means to learn nonlinear features from observed data, with low-level features in the bottom

layers and more abstract, high-level features in the top layers. DL has shown outstanding results in many applications, e.g., in image processing [35] and especially in automated speech recognition [36–38].

Some FL approaches have been extended for the use in time-sequential data, e.g., for feature analysis in videos. Slow Feature Analysis (SFA), which can be considered as an extension of ICA, assumes that the relevant information of a time varying signal is contained in a slowly varying latent signal [39]. This latent signal can then be considered as the features of the observed signal. In [40], SFA has been extended to decision-making problems, referred to as Contingent Feature Analysis. In this approach, features are sought for that explain the high variance of the temporal derivatives of the observed states, if the agent performs an action. A different approach based on DL has been taken in [41], where the independent subspace analysis algorithm [42] has been extended such that features are directly learned from video material, resulting in highly accurate action recognition. Recent research has demonstrated the performance of DL also in decision-making problems [43, 44].

Especially in image formation, capturing the data at Nyquist rate and then transforming it into a low-dimensional representation is inefficient, as costly acquired data is literally thrown away. A framework that allows for the acquisition of data below the Nyquist rate is given by Compressive Sensing (CS) [45, 46]. In CS, one assumes that the data that shall be captured can be sparsely represented in a certain domain. Under certain conditions, the data can be perfectly reconstructed from the few captured samples. CS can be also understood as capturing a low-dimensional representation directly at the sensing level. However, finding suitable sparse transformation and measurement matrices, which can be considered as a linear feature transform, is generally challenging. While the literature on CS is exhaustive, only few work is concerned with learning the dictionary for CS, e.g., [47, 48].

The state of the art of FL is currently dominated by DL. Though these models show excellent results, they possess significant drawbacks. First, it is not easy to incorporate prior assumptions in deep architectures. If FL is used for data analysis with the aim to learn about the structure of the observed data, DL does not present a suitable approach as the learned representations are often difficult to interpret. Second, training is highly challenging since these architectures are able to represent highly complex nonlinear models. Thus, there is no guarantee that (globally) optimal parameters are learned with respect to classification accuracy. Further, despite the compactness of these models, a huge number of parameters has to be learned, requiring large amounts of data and vast computation facilities. Though costs for computation power has significantly decreased with the availability of massively parallel computer architectures such as

graphics cards and clusters, computation time is still an issue for practical data analysis. More important than that, for many problems we do not have access to large amounts of observations for training the model. Instead, we often have an intuition about the characteristics of the data, which can be exploited to constrain the model. By this, the flexibility of the model can be reduced and, hence, the number of parameters that need to be learned, making inference more efficient in the sense of data and computation efficiency.

1.2 Contributions

We propose different feature selection and learning techniques with applications to hyperspectral imaging and decision-making. For hyperspectral imaging applications, we consider a feature selection scheme for classification and a low-dimensional acquisition method for direct classification of the observed data. Moreover, a Bayesian approach for unmixing hyperspectral images based on BFL is proposed. Finally, we extend the BFL framework to learn feature representations in decision-making problems.

In summary, the contributions of this thesis are as follows:

- **Band selection for hyperspectral imaging**

We provide a simple yet effective algorithm that selects relevant bands of a hyperspectral image which serve as features for classification. Operating in the subspace of the observed data, we provide means to estimate the number of relevant bands.

- **Classification based on sparsely acquired hyperspectral image data**

Capturing hundreds of bands and removing many of them during band selection is resource-inefficient. Thus, we present an acquisition scheme that directly captures the relevant information of the scene. The proposed framework is based on CS but skips the costly reconstruction of the data. Instead, the compressively sensed data is considered as a low-dimensional representation and is directly classified. To improve the classification accuracy, we show how the sensing and transformation matrices can be learned from training data.

- **Inferring the latent number of endmembers, the endmembers, and their fractional abundances in hyperspectral images**

Based on BFL, we consider an algorithm that decomposes the elements of a hyperspectral image into its endmembers and their fractional abundances. As the

number of latent endmembers is rarely known in practice, we present a Bayesian nonparametric extension that is also able to estimate the number of endmembers.

- **Feature-based decision-making**

We adapt and extend BFL to a Bayesian nonparametric formulation for behavior analysis in the context of decision-making. We assume that the observed agent makes its decision based on latent features of the observed state. These features can then be interpreted as causes that influenced the agent to make the observed decision. Our goal is to learn this representation, which provides insights into the reasoning of the agent. Additionally, the proposed model allows for a compact state representation that can be used for efficient prediction of actions for new observations.

1.3 Publications

The following publications have been produced during this doctoral project.

Internationally Refereed Journal Articles

- J. Hahn and A. M. Zoubir, “Bayesian Nonparametric Feature and Policy Learning for Decision Making”, submitted to *Pattern Recognit*, 2016
- J. Hahn and A. M. Zoubir, “Bayesian Nonparametric Unmixing of Hyperspectral Images”, submitted to *IEEE Trans Geosci Remote Sens*, 2016
- C. Debes, A. Merentitis, R. Heremans, J. Hahn, N. Frangiadakis, T. van Kasteren, W. Liao, R. Bellens, A. Pizurica, S. Gautama, W. Philips, S. Prasad, Q. Du, and F. Pacifici, “Hyperspectral and LiDAR Data Fusion: Outcome of the 2013 GRSS Data Fusion Contest” *IEEE J Sel Topics Appl Earth Observ*, Vol 7, No 6, pp. 2405-2418, Mar. 2014
- J. Hahn, C. Debes, M. Leigsnering, and A. M. Zoubir, “Compressive Sensing and Adaptive Direct Sampling in Hyperspectral Imaging”, *Digit Signal Process*, Vol 26, No 3, pp 113-126, Mar. 2014

Internationally Refereed Conference Papers

- J. Hahn and A. M. Zoubir, “Risk-Sensitive Decision Making via Constrained Expected Returns”, *Proc. IEEE Int Conf Acoust Speech Signal Process. (ICASSP) 2016* in Shanghai, China, pp. 2569-2573, Mar. 2016
- S. Shukanov, A. Merentitis, C. Debes, J. Hahn, and A. M. Zoubir, “Bootstrap-Based SVM Aggregation for Class Imbalance Problems”, *Proc. Eur Signal Process Conf (EUSIPCO) 2015* in Nice, France, pp. 165-169, Sept. 2015
- J. Hahn and A. M. Zoubir, “Inverse Reinforcement Learning Using Expectation Maximization in Mixture Models”, *Proc. IEEE Int Conf Acoust Speech Signal Process (ICASSP) 2015* in Brisbane, Australia, pp. 3721-3725, Apr. 2015
- J. Hahn, S. Rosenkranz, and A. M. Zoubir, “Adaptive Compressed Classification for Hyperspectral Imagery”, *Proc. IEEE Int Conf Acoust Speech Signal Process (ICASSP) 2014* in Florence, Italy, pp. 1020-1024, May 2014
- V. Kumar, J. Hahn, and A. M. Zoubir, “Band Selection for Hyperspectral Images Based on Self-Tuning Spectral Clustering”, *Proc. Eur Signal Process Conf (EUSIPCO) 2013* in Marrakech, Morocco, pp. 1-5, Sept. 2013

1.4 Organization of the Thesis

The thesis is structured as follows. In Chapter 2, an introduction to machine learning theory is given. Further, basic feature learning and classification approaches are presented that are used throughout the thesis.

In Chapter 3, we first introduce a band selection algorithm for classification. Since the bands serve as features for the classifier, this approach can be considered as a feature selection algorithm tailored to hyperspectral imaging. In the same chapter, we present an alternative approach that aims at capturing image data in a low-dimensional representation directly by means of the CS framework. Since we are interested in the classification of the image instead of the actual hyperspectral image, we skip the costly reconstruction step and consider the captured data as a low-dimensional feature representation that is directly classified. The proposed method is adaptive in the sense that the measurement and basis matrices are learned from training data.

In Chapter 4, we present a Bayesian nonparametric framework for hyperspectral unmixing. This framework is able to jointly infer the endmembers present in the scene,

their fractional abundances and, in contrast to many existing approaches, the number of the endmembers. The proposed algorithm is evaluated with simulated as well as real data.

In Chapter 5, the Bayesian feature learning framework is extended and applied to a decision-making problem. We assume that the agent, whose states and actions can be observed, acts upon latent features of the observations. Thus, each feature indicates a certain policy, leading to a mixture of policies the agent follows. In particular, we are interested in learning the features and their policies, as they provide a deeper understanding of the observed behavior as well as a compact representation of the states, allowing for efficient prediction of actions for new, previously unobserved states.

Finally, in Chapter 6, conclusions are drawn and an outlook for future perspectives is given.

Chapter 2

Machine and Feature Learning Fundamentals

In this chapter, we give an introduction to computational learning theory. Based on the presented concepts, inference techniques and methods for dimensionality reduction and classification are explained which are used throughout this thesis.

2.1 Computational Learning Theory

Computational learning theory provides means to understand and to model learning problems, and enables performance analysis of developed algorithms. There exist many different philosophies in the field of learning theory, though many theories are based on similar ideas and possess common assumptions. In this thesis, we rely on statistical learning theory [49] and Bayesian inference to model the given learning problems.

2.1.1 Statistical Learning Theory

Statistical learning theory describes the act of learning from experience to draw conclusions about future observations, referred to as predictions [49, 50]. Essentially, this can be described as learning a mapping, $f_{\theta} : \mathbf{z} \mapsto y$, where $\mathbf{z} \in \mathcal{Z}^{D \times 1}$ is the observed data sample of dimension D and $\theta \in \Theta$ describes the parameters of the function f_{θ} . The parameters are learned from previously observed data \mathcal{D} and are used to predict the label $y \in \mathcal{Y}$. If \mathcal{Y} denotes a finite set, we refer to problem as *classification*, while in contrast, if \mathcal{Y} is an infinite set, we refer to the problem as *regression*. This scheme is illustrated in Fig. 2.1.

For learning θ , a loss function, \mathbb{L} , is defined, which penalizes incorrect predictions. During learning, the expected risk shall be minimized, which is the expectation of the loss function,

$$\mathbb{E}_{\text{Risk}}(f) = \int_{\mathcal{Z}} \int_{\mathcal{Y}} \mathbb{L}(f_{\theta}(\mathbf{z}), y) p(\mathbf{z}, y) d\mathbf{z} dy, \quad (2.1)$$

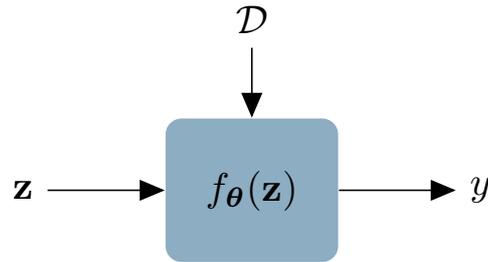


Figure 2.1. Learning refers to estimating the parameters, θ , of function f_{θ} from previously observed data \mathcal{D} . A label, y , for observation data, \mathbf{z} , can then be predicted based on θ .

where $p(\mathbf{z}, y)$ denotes the joint distribution of the observations, \mathbf{z} , and their labels, y . Since this distribution is generally unknown, it is approximated by the samples contained in the training set $\mathcal{D}_T = \{(\mathbf{z}_{T,1}, y_{T,1}), \dots, (\mathbf{z}_{T,N_T}, y_{T,N_T})\}$, yielding the empirical risk,

$$\mathbb{E}_{\text{Emp}}(f) = \frac{1}{N_T} \sum_{n=1}^{N_T} \mathbb{L}(f_{\theta}(\mathbf{z}_{T,n}), y_{T,n}). \quad (2.2)$$

The empirical risk in Eq. (2.2) converges to the true risk in Eq. (2.1) for an infinite number of training samples, $N_T \rightarrow \infty$. In practice, however, the size of the training set is often limited to a few samples, such that the empirical risk only poorly approximates the true risk. Then, minimizing the empirical risk does not guarantee to provide good estimates of the parameters. This is related to *overfitting*, which describes the problem that the parameters θ are well adapted to the training set, but do not generalize well to unobserved data. Structural risk minimization aims at solving this problem by finding a “tradeoff between the quality of the approximation and the complexity of the approximating function” [49] and has resulted, e.g., in the development of the Support Vector Machine (SVM) [51]. The SVM is explained in Section 2.4.3.

Learning problems where predictions are part of the training data set are referred to as *supervised* learning tasks and are well understood in the framework of statistical learning theory. If the training data set, \mathcal{D}_T , only consists of the observations $\mathbf{z}_1, \dots, \mathbf{z}_{N_T}$, we refer to the problem as an *unsupervised* learning task. There are other types of learning problems, such as online or reinforcement learning, that are not considered in this thesis. Though the observations in the decision-making problem in Chapter 5 underlie a Markov Decision Process (MDP), the problem we consider can be reduced to a supervised one. For completeness, we will briefly introduce concepts and models for decision-making in Chapter 5.

2.1.2 Bayesian Inference

A different, yet similar paradigm to the statistical learning framework is statistical inference [52, 53]. Statistical inference provides means for understanding both supervised and unsupervised problems by building models that aim at explaining the observed data.

In this thesis, we mainly rely on Bayesian inference, a subarea of statistical inference, that we consider in particular in Chapters 4 and 5. Bayesian inference is a universal tool which provides means for modeling various problems and inferring solutions. Prior assumptions and constraints can be easily incorporated, making it interesting to be used in feature learning [17]. In the following, we briefly introduce the fundamentals of Bayesian inference.

A key assumption of Bayesian inference is that observed data cannot be explained by fixed parameters. Instead, the parameters are considered as variables, in the following denoted by $\boldsymbol{\theta}$. Placing a prior distribution over the variables, $p(\boldsymbol{\theta})$, the goal is to derive the posterior distribution, $p(\boldsymbol{\theta} | \mathcal{D})$, over the possible parameters, providing information about how probable a certain parameter is given the set of observations, \mathcal{D} . Using Bayes' theorem, we can express the posterior in terms of the likelihood, $p(\mathcal{D} | \boldsymbol{\theta})$, and the prior, $p(\boldsymbol{\theta})$,

$$\underbrace{p(\boldsymbol{\theta} | \mathcal{D})}_{\text{posterior}} = \frac{p(\mathcal{D} | \boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathcal{D})} \propto \underbrace{p(\mathcal{D} | \boldsymbol{\theta})}_{\text{likelihood}} \underbrace{p(\boldsymbol{\theta})}_{\text{prior}}. \quad (2.3)$$

To find an expression for the posterior $p(\boldsymbol{\theta} | \mathcal{D})$, the factorization in Eq. (2.3) is exploited. Prior knowledge or general assumptions on the parameters can be incorporated into the prior. The normalization, referred to as evidence, given by $p(\mathcal{D})$, is usually not considered as it is independent of the parameters. As the posterior is often not analytically tractable, one has to resort to approximate inference techniques. This is detailed in Section 2.2.

For prediction, Bayesian inference averages over all possible explanations represented by $\boldsymbol{\theta}$. Given the training data, \mathcal{D} , and an observation, \mathbf{z}^* , we want to infer the corresponding value y^* , which is achieved by means of the *posterior predictive* distribution:

$$\begin{aligned} p(y^* | \mathbf{z}^*, \mathcal{D}) &= \int_{\Theta} p(y^*, \boldsymbol{\theta} | \mathbf{z}^*, \mathcal{D}) \, d\boldsymbol{\theta} \\ &= \int_{\Theta} p(y^* | \boldsymbol{\theta}, \mathbf{z}^*) p(\boldsymbol{\theta} | \mathbf{z}^*, \mathcal{D}) \, d\boldsymbol{\theta}. \end{aligned}$$

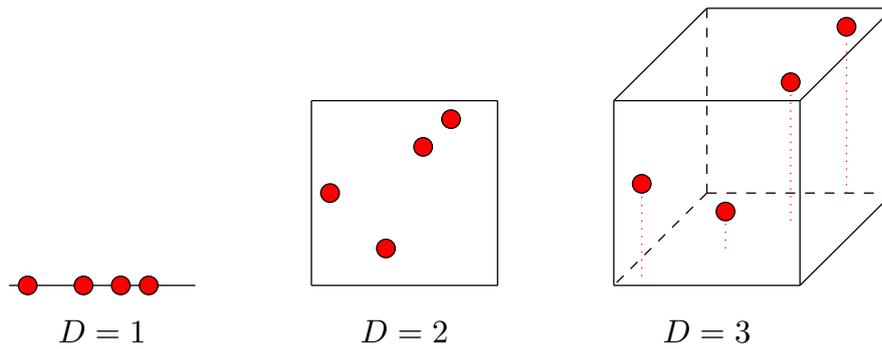


Figure 2.2. Illustration of the exponential growth of the volume of the space with increasing dimension D . As the number of samples (red dots) is kept constant the coverage of the space becomes highly sparse.

Here, we assume that the parameters, θ , contain all relevant information such that y^* is independent of \mathcal{D} given θ .

In general, θ depends on the training data, \mathcal{D} , and the new observation, \mathbf{z}^* . Thus, the posterior in Eq. (2.3) has to be estimated for every new observation. However, if the training samples sufficiently characterize θ , the influence of \mathbf{z}^* on θ is relatively small such that the dependency on the new observations can be ignored, reducing prediction time significantly.

2.1.3 Curse of Dimensionality

Many machine learning and statistical inference techniques show poor performance with increasing dimensionality of the data. The *curse of dimensionality*, also known as Hughes effect [5], explains the fact why the required number of training samples grows exponentially with the dimensionality of the data to counteract this phenomenon [54]. As illustrated in Fig. 2.2, with growing dimensionality, the observation space increases exponentially, leading to a sparse coverage by training samples.

Many signal processing and machine learning algorithms suffer from this fact. The training samples are hardly able to represent the distribution of the data, which makes learning the underlying structure difficult and results in a poor prediction performance. This effect can be easily explained by means of the statistical learning framework, as the empirical risk, Eq. (2.2), becomes a poor approximation of the true risk, Eq. (2.1).

For this reason, it is often desirable to find a low-dimensional representation of the data, which can be understood as a feature representation. Note that, however, reducing the

dimensionality to a minimum does not always result in a better performance. A good counterexample can be found in classification, where a high-dimensional representation often helps to separate the classes.

2.2 Approximate Inference Techniques

In many applications, the posterior, which is sought for in Bayesian inference, is analytically intractable. We encounter this problem in the hyperspectral unmixing model and the decision-making problem in Chapters 4 and 5, respectively.

For this reason, techniques for approximating the posterior are required. On the one hand, we can try to express the posterior by a simpler distribution that allows for an analytic solution as in variational methods, c.f. [55]. On the other hand, we can represent the posterior by samples, known as Monte Carlo techniques, c.f. [56]. Sampling based approaches have the advantage that they are able to perfectly represent the distribution if, in the limit, an infinite number of samples is given. The main drawback is that, usually, sampling is more time consuming compared with variational techniques [53]. Though variational approaches are often significantly faster, we only consider sampling based approaches as the approximations in the variational framework can introduce a bias. Well-known sampling algorithms are the Metropolis-Hastings algorithm and the Gibbs sampler, which are introduced in the following sections.

2.2.1 Metropolis and Metropolis-Hastings Sampling

Consider the problem of generating samples of the random variable $\boldsymbol{\theta} \in \Theta$, where the probability density function (pdf) of $\boldsymbol{\theta}$, referred to as target pdf, can be written as

$$p(\boldsymbol{\theta}) = \frac{1}{Z_{\boldsymbol{\theta}}} \tilde{p}(\boldsymbol{\theta}),$$

with $\tilde{p}(\boldsymbol{\theta})$ denoting a factor dependent on $\boldsymbol{\theta}$. In general, sampling from $p(\boldsymbol{\theta})$ can be challenging, especially if the normalization constant $Z_{\boldsymbol{\theta}}$ is unknown. Metropolis *et al.* [57] proposed an iterative method for sampling from $p(\boldsymbol{\theta})$ using a proposal distribution, $q(\boldsymbol{\theta}^{(t)} | \boldsymbol{\theta}^{(t-1)})$, one can easily sample from, with $t, t = 1, \dots, N_S$, denoting the t th sample. The proposal $\boldsymbol{\theta}^{(t)}$ generally depends on the previously generated sample $\boldsymbol{\theta}^{(t-1)}$, where the first sample can be initialized arbitrarily. Hence, this sampler forms a Markov chain.

In each iteration, the t th proposed sample $\boldsymbol{\theta}^{(t)}$ is accepted with probability $P_M = \min(r_M, 1)$, with acceptance ratio

$$r_M = \frac{\tilde{p}(\boldsymbol{\theta}^{(t)})}{\tilde{p}(\boldsymbol{\theta}^{(t-1)})}.$$

If the sample is rejected, the previous sample is kept, i.e., $\boldsymbol{\theta}^{(t)} \leftarrow \boldsymbol{\theta}^{(t-1)}$. Note that several iterations, known as the *burn-in* phase, are required until samples from the target distribution are drawn.

A key limitation of this algorithm is the assumption that the proposal distribution is symmetric, i.e., $q(\boldsymbol{\theta}^{(t)} | \boldsymbol{\theta}^{(t-1)}) = q(\boldsymbol{\theta}^{(t-1)} | \boldsymbol{\theta}^{(t)})$. The Metropolis-Hastings (MH) algorithm [58] generalizes the Metropolis algorithm by incorporating the probabilities of the proposals into the modified acceptance ratio, r_{MH} , with

$$r_{MH} = \frac{\tilde{p}(\boldsymbol{\theta}^{(t)})q(\boldsymbol{\theta}^{(t-1)} | \boldsymbol{\theta}^{(t)})}{\tilde{p}(\boldsymbol{\theta}^{(t-1)})q(\boldsymbol{\theta}^{(t)} | \boldsymbol{\theta}^{(t-1)})}.$$

It can easily be seen that if the proposal distribution is symmetric, the MH algorithm is equal to the Metropolis algorithm. Both algorithms perform a random walk and the samples $\boldsymbol{\theta}^{(t)}, t = 1, \dots, N_S$, are distributed according to the target pdf for $N_S \rightarrow \infty$.

The samples generated by Metropolis sampling are usually highly correlated for mainly two reasons. First, the sampler may repeat samples due to high rejection rates. Second, as the sampler performs a random walk, the differences between the samples might be small, yielding very similar samples. The first problem can be alleviated by choosing a proposal distribution that proposes samples close to previous sample, which, however, increases the second problem. The authors in [59] found that a rejection rate of 0.234 yields a good trade-off. Ideally, we choose a distribution that is very similar to the target pdf. This is the key idea in Gibbs Sampling, which is introduced in the next section.

2.2.2 Gibbs Sampling in a Directed Graphical Model

Sampling high-dimensional variables by means of Metropolis sampling can be highly time consuming due to high rejection rates. We can alleviate this problem by considering each random variable separately, given the other variables. If the conditional of each variable can be derived analytically and be sampled from easily, we may use the conditional as proposal distribution. It can be shown [60] that this results in an acceptance ratio of one such that all samples are accepted. This method is referred

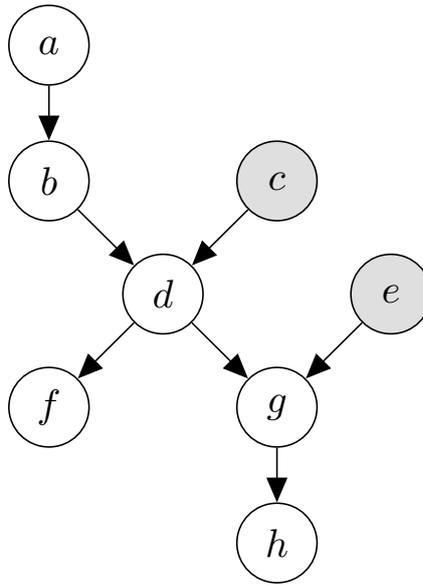


Figure 2.3. Illustration of a graphical model with variables a – h . As an example, the Markov blanket of d consists of the variables b, c, f, g and e . Observed variables are generally shaded in gray, e.g., c and e .

to as Gibbs Sampling [61] and is frequently used in Bayesian statistics, especially in Bayesian networks. Bayesian networks describe the structure of a multivariate distribution in form of the dependencies between the variables and can be represented by a Directed Acyclic Graph (DAG), as opposed to Markov Random Fields (MRFs).

The conditionals, required for Gibbs sampling, are easily derived from a DAG as they are given by the *Markov blanket* of the variables [62]. Thus, the conditional of the d th element of the random vector $\boldsymbol{\theta}$ can be expressed by

$$p(\theta_d | \boldsymbol{\theta}_{\setminus d}) \propto p(\theta_d | \text{pa}(\theta_d)) \prod_{j=\text{ch}(\theta_d)} p(\theta_j | \text{pa}(\theta_j)),$$

where $\boldsymbol{\theta}_{\setminus d}$ denotes all variables except the d th. The operator $\text{pa}(\theta)$ denotes the parents of variable θ and $\text{ch}(\theta)$ its children. An example of a graphical model is depicted in Fig. 2.3, illustrating observed and latent variables and the Markov blanket.

The models we investigate in Chapters 4 and 5 belong to the class of Bayesian networks. For this reason, we consider only Metropolis-Hasting and Gibbs sampling in this chapter. However, there exist many other sampling techniques such as slice sampling [63] or Metropolis-Hamilton-Dynamics [64] that provide efficient means for drawing samples from the target distribution.

2.2.3 Parallel Tempering

Gibbs sampling is known for getting easily trapped in modes of the joint distribution. In this case, the distribution is not correctly represented by the samples, leading to incorrect inference. This can be detected by running multiple samplers independently in parallel and checking the results for convergence. In order to sample correctly from a multi-modal distribution, Parallel Tempering (PT) [65–67] can be utilized. The key idea, similar as in simulated annealing [61], is to augment the target pdf with another variable, the temperature T_{PT} . Effectively, higher temperatures will smooth out the modes so that other states of the chain, i.e., other realizations of the (modified) distribution, become more probable. Hence, several chains are run in parallel and are exchanged using a Metropolis step.

The algorithm is derived as follows: let the target pdf of the variable $\boldsymbol{\theta} \in \Theta$ be

$$p(\boldsymbol{\theta}) = \frac{1}{Z_{\boldsymbol{\theta}}} \exp\{-U(\boldsymbol{\theta}) - V(\boldsymbol{\theta})\},$$

with the two energy terms $U(\boldsymbol{\theta})$, $U : \Theta \rightarrow \mathbb{R}$, and $V(\boldsymbol{\theta})$, $V : \Theta \rightarrow \mathbb{R}$. We augment $U(\boldsymbol{\theta})$ with the temperature $T_{\text{PT}} \in \{T_{\text{PT}}^{(1)}, \dots, T_{\text{PT}}^{(N_{\text{PT}})}\}$ where N_{PT} denotes the number of temperature levels. As an example, $U(\boldsymbol{\theta})$ could describe the energy term of a likelihood, whereas $V(\boldsymbol{\theta})$ could be the energy term of a prior. Thus, augmenting $U(\boldsymbol{\theta})$ only would be equivalent to augmenting the likelihood. For convenience, the temperature levels are ordered, such that

$$T_{\text{PT}}^{(1)} = 1 < T_{\text{PT}}^{(2)} < \dots < T_{\text{PT}}^{(N_{\text{PT}})}.$$

The pdf augmented with the i th temperature is given as

$$p(\boldsymbol{\theta} | T_{\text{PT}}^{(i)}) = \frac{1}{Z_{\boldsymbol{\theta}, T_{\text{PT}}^{(i)}}} \exp\left\{-\frac{1}{T_{\text{PT}}^{(i)}} U(\boldsymbol{\theta}) - V(\boldsymbol{\theta})\right\}.$$

The probability, $P_{\text{PT}}^{(ij)}$, of accepting a swap of the i th and the j th chain is

$$P_{\text{PT}}^{(ij)} = \min(1, r_{\text{PT}}^{(ij)}),$$

with acceptance ratio $r_{\text{PT}}^{(ij)}$,

$$\begin{aligned} r_{\text{PT}}^{(ij)} &= \frac{p(\boldsymbol{\theta}_j | T_{\text{PT}}^{(i)})p(\boldsymbol{\theta}_i | T_{\text{PT}}^{(j)})}{p(\boldsymbol{\theta}_i | T_{\text{PT}}^{(i)})p(\boldsymbol{\theta}_j | T_{\text{PT}}^{(j)})} \\ &= \frac{\exp\left\{-\frac{1}{T_{\text{PT}}^{(j)}} U(\boldsymbol{\theta}_i) - \frac{1}{T_{\text{PT}}^{(i)}} U(\boldsymbol{\theta}_j)\right\}}{\exp\left\{-\frac{1}{T_{\text{PT}}^{(j)}} U(\boldsymbol{\theta}_j) - \frac{1}{T_{\text{PT}}^{(i)}} U(\boldsymbol{\theta}_i)\right\}} \\ &= \exp\left\{\left(\frac{1}{T_{\text{PT}}^{(j)}} - \frac{1}{T_{\text{PT}}^{(i)}}\right)(U(\boldsymbol{\theta}_j) - U(\boldsymbol{\theta}_i))\right\}. \end{aligned}$$

Samples of the target pdf are taken from the first chain with $T_{PT} = 1$ only. Theoretical results on how to find a suitable number of temperature levels, N_{PT} , and how to set the temperatures can be derived only for simple cases [66]. In practice, one often resorts to setting these parameters in a trial and error approach.

2.3 Feature Learning and Low-Dimensional Representations

Feature learning enables the automatic extraction of relevant information from high-dimensional data and to transform it into a low-dimensional representation for efficient data analysis.

There are also approaches that transform the observations into a high-dimensional representation, e.g., in kernel machines, c.f. Section 2.4.3. High-dimensional representations are mainly used to improve the separability of the observations as required for classification, for example. However, high dimensional representations are often difficult to interpret and to visualize. For these reasons, and in order to alleviate the curse of dimensionality, in this thesis, we generally understand feature representations as low-dimensional representations of the observed data.

In this section, we revisit methods for acquiring and learning a low-dimensional representation of the observed data. These approaches present the basis for the techniques developed in the following chapters. First, we introduce Compressive Sensing (CS) [45, 46], which allows for sparse acquisition of the data. Second, we present a Bayesian approach for learning low-dimensional representations, referred to as Bayesian Feature Learning (BFL). A great advantage of the presented Bayesian approach compared with many existing approaches is the possibility to infer the number of features from the data. In other methods, this parameter often has to be set in a trial-and-error approach.

2.3.1 Feature Models

A key difference between CS and BFL is how the features are related to the observations. In the Bayesian formulation, we consider a generative model, i.e., the observations are composed of the features, denoted by \mathbf{F}_{Gen} , and their low-dimensional

representation, \mathbf{s}_{Gen} ,

$$\mathbf{z} = \mathbf{F}_{\text{Gen}}\mathbf{s}_{\text{Gen}}. \quad (2.4)$$

In contrast, CS is related to a projection-based approach, where we interpret the rows of the projection matrix, \mathbf{F}_{Dis} , as features,

$$\mathbf{s}_{\text{Dis}} = \mathbf{F}_{\text{Dis}}\mathbf{z}, \quad (2.5)$$

with \mathbf{s}_{Dis} denoting the projected observations. As this model yields the low-dimensional representation given the features directly, we refer to this approach as *discriminative* feature learning. In contrast, in the Bayesian model, the representations are part of the generative model. Thus, we refer to the model described in Eq. (2.4) as *generative* feature learning. In the following, we do not differentiate between these models as this will be clear from the context.

2.3.2 Low-Dimensional Data Acquisition Using Compressive Sensing

In order to capture a signal in a low-dimensional representation directly, the CS framework can be used [45, 46]. For this purpose, the signal is required to be sparse in a certain domain. Let $\mathbf{z} = (z_1, \dots, z_D)^T$ denote a signal of length D that can be sparsely represented by means of an orthonormal transformation matrix, $\mathbf{\Psi}$, which describes a $D \times D$ orthonormal basis such that

$$\mathbf{z} = \mathbf{\Psi}\mathbf{x}, \quad (2.6)$$

where $\mathbf{x} = (x_1, \dots, x_D)^T$ denotes the K_S -sparse coefficient vector. The K_S -sparse property states that only $K_S \ll D$ coefficients are non-zero.

In contrast to conventional sensing following the Nyquist-Shannon sampling theorem [68], not all D samples of the signal \mathbf{z} need to be captured when using CS. The observed vector $\mathbf{s} = (s_1, \dots, s_K)^T$ is captured using only $K < D$ measurements, i.e.,

$$\mathbf{s} = \mathbf{\Phi}\mathbf{z} + \mathbf{e} = \mathbf{\Phi}\mathbf{\Psi}\mathbf{x} + \mathbf{e},$$

where $\mathbf{\Phi} = [\phi_1 \ \dots \ \phi_K]^T$ denotes the $K \times D$ measurement matrix and the random vector \mathbf{e} describes additive Gaussian noise. For accurate reconstruction, the measurement and basis matrices need to be incoherent, which is stated by the Restricted Isometry Property (RIP) [69]. It has been shown that a good choice of $\mathbf{\Phi}$ is given

by Gaussian and Bernoulli random matrices [46, 70], which fulfill the RIP with high probability.

For the estimation of \mathbf{z} , an under-determined system of equations needs to be solved, which is possible thanks to the sparsity of \mathbf{x} . Consequently, only K , with $K_S \leq K \ll D$, measurements are actually required for a highly accurate reconstruction [71]. Thus, the number of samples can be significantly reduced.

The sparsity of \mathbf{x} is exploited during reconstruction by employing a regularization term. Ideally, the l_0 -norm is considered, resulting however in an infeasible NP-hard problem [72]. In [45, 46], it is shown that minimizing the l_1 -norm yields a good approximation, i.e., the reconstructed signal $\hat{\mathbf{x}}$ is obtained by

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}} \|\mathbf{x}\|_1 \quad \text{s.t.} \quad \|\Phi\Psi^T\mathbf{x} - \mathbf{s}\|_2 \leq \epsilon, \quad (2.7)$$

where ϵ is a threshold for the tolerated reconstruction error. Given the estimate $\hat{\mathbf{x}}$, an estimate of the signal $\hat{\mathbf{z}}$ can be obtained using Eq. (2.6). The optimization problem formulated in Eq. (2.7) can be efficiently solved thanks to the convex shape of the function to be minimized. For this purpose, many algorithms and software packages have been presented, e.g., l_1 -magic [73] and SpaRSA [74].

2.3.3 Bayesian Feature Learning

The BFL framework provides a flexible model for inferring the latent structure of the observed data, $\mathbf{z}_n \in \mathcal{Z}^{D \times 1}$, $n = 1, \dots, N_z$. For tractability, we focus on a linear feature model as in [15, 17, 75], which states that each observation is composed of a linear mixture of features,

$$\mathbf{z}_n = \sum_{k=1}^K \mathbf{f}_k s_{n,k} + \mathbf{e}_n, \quad n = 1, \dots, N_z, \quad (2.8)$$

with feature coefficient vector $\mathbf{s}_n = (s_{n,1}, \dots, s_{n,K})^T \in \mathcal{S}^{K \times 1}$ and K feature vectors, $\mathbf{f}_k \in \mathcal{F}^{D \times 1}$, $k = 1, \dots, K$. The domain of the observations, \mathcal{Z} , the feature coefficients, \mathcal{S} , and the features, \mathcal{F} , clearly depend on the application. Thus, the prior distributions for the coefficients and features have to be chosen accordingly.

For example, in Bayesian Non-negative Matrix Factorization (NMF) [17], both the feature coefficients and vectors are assumed to be positive real-valued, which can be expressed by modeling the priors for \mathbf{s}_n , $n = 1, \dots, N_z$, and \mathbf{f}_k , $k = 1, \dots, K$, as Gamma

distributions. The observation likelihood is modeled with a Gaussian distribution, i.e., the noise term \mathbf{e}_n , $n = 1, \dots, N_z$, is assumed to follow a Gaussian distribution. As the posterior distribution cannot be derived analytically, the authors resort to Gibbs sampling for inference [17].

A key element of BFL is the assumption that the number of features, K , is known, which, in practice, rarely holds. However, BFL can be extended to a nonparametric version in which the number of features is inferred from the data [15, 75]. We refer to this framework as Bayesian Nonparametric Feature Learning (BNFL). In BNFL, we consider a modified feature model, where we assume that the features are composed of the feature weights, $\mathbf{W} \in \mathcal{F}^{D \times K}$, and binary feature activations, $\mathbf{A} \in \{0, 1\}^{D \times K}$, [75]

$$\mathbf{F} = \mathbf{A} \odot \mathbf{W}, \quad (2.9)$$

with \odot denoting the element-wise Hadamard product and $\mathbf{F} = [\mathbf{f}_1 \ \dots \ \mathbf{f}_K]$ the feature matrix. Again, the domain and, hence, the prior for \mathbf{W} , depends on the given task. The prior for \mathbf{A} is given by an Indian Buffet Process (IBP). The IBP assumes an infinite number of features, where only a finite number is present in the data. Sparsity is a fundamental assumption made in the derivation of the IBP, resulting in a finite number of features in the realizations of the variable. As will be explained in the next section, though the IBP promotes sparse feature realizations, dense realizations can be also drawn.

Note that the model described in Eqs. 2.8 and 2.9 is similar to a spike and slab model [24, 25] when computing the marginal over the feature activations [76, 77].

2.3.3.1 Indian Buffet Process

The Indian Buffet Process (IBP) [15] describes a model for sampling a sparse binary feature matrix, assuming an infinite number of features. In the following, we focus on the two-parameter generalization that provides means to model dense activations and, thus, alleviates the sparsity assumption [78]. The full derivation of the IBP is given in [15, 79].

For a finite number of features, K , the sums over the columns of the feature activation matrix $\mathbf{A} \in \{0, 1\}^{D \times K}$ are assumed to follow i.i.d. Binomial distributions, with D denoting the dimension of the features. Placing a Beta prior with hyperparameters

$\frac{\alpha_a \beta_a}{K}$ and β_a over the parameter, θ_a , of the Bernoulli distribution and marginalizing over θ_a yields a Beta-Binomial distribution,

$$\begin{aligned}
P(\mathbf{A} | \alpha_a, \beta_a) &= \prod_{k=1}^K \int_0^1 P(\mathbf{a}_k | \theta_a) p(\theta_a | \frac{\alpha_a \beta_a}{K}, \beta_a) d\theta_a \\
&= \prod_{k=1}^K \int_0^1 \text{Bin}_{m_k}(\theta) \text{Beta}_\theta\left(\frac{\alpha_a \beta_a}{K}, \beta_a\right) d\theta_a \\
&= \prod_{k=1}^K \frac{B(m_k + \frac{\alpha_a \beta_a}{K}, D - m_k + \beta_a)}{B(\frac{\alpha_a \beta_a}{K}, \beta_a)} \\
&= \prod_{k=1}^K \text{BetaBin}_{m_k}\left(\frac{\alpha_a \beta_a}{K}, \beta_a\right),
\end{aligned} \tag{2.10}$$

where \mathbf{a}_k denotes the k th column of \mathbf{A} , m_k counts the number of ones in \mathbf{a}_k and $B(m, n)$ is the Beta function with arguments m and n .

The distribution in Eq. (2.10) describes unsorted binary matrices such that different binary matrices may represent equivalent feature activations. In particular, permutations of the columns of \mathbf{A} describe the same feature representations, $[\mathbf{A}]$. Thus, normalizing Eq. (2.10) yields the distribution of feature representations,

$$P([\mathbf{A}] | \alpha_a, \beta_a) = \frac{1}{Z_{\text{IBP}}} P(\mathbf{A} | \alpha_a, \beta_a),$$

with normalization Z_{IBP} ,

$$Z_{\text{IBP}} = \left(\prod_{\mathbf{h} \in \{0,1\}^D} K_{\mathbf{h}} \right),$$

where $K_{\mathbf{h}}$ denotes the number of occurrences of the binary vector $\mathbf{h} \in \{0,1\}^D$ in the columns of \mathbf{A} and the brackets $()$ indicate the binomial coefficient. Ultimately, as we want to model an infinite number of features, we consider the limit for $K \rightarrow \infty$ [15, 78],

$$\begin{aligned}
P([\mathbf{A}_\infty] | \alpha_a, \beta_a) &:= \lim_{K \rightarrow \infty} P([\mathbf{A}] | \alpha_a, \beta_a) \\
&= \frac{(\alpha_a \beta_a)^{K_+}}{\prod_{\mathbf{h} \in \{0,1\}^D \setminus \mathbf{0}} K_{\mathbf{h}}!} \exp\{-\bar{K}_+\} \prod_{k=1}^{K_+} B(m_{\infty,k}, D - m_{\infty,k} + \beta_a),
\end{aligned} \tag{2.11}$$

where K_+ denotes the number of active features, $\bar{K}_+ = \alpha_a \sum_{d=1}^D \frac{\beta_a}{\beta_a + d - 1}$ is the expected number of active features and $\mathbf{0}$ is the null vector [15, 78]. The sum over the k th column of \mathbf{A}_∞ is denoted by $m_{\infty,k}$. The derivation for the limit in Eq. (2.11) can be found in [79]. From Eq. (2.11), we can further conclude that the number of active features, K_+ , is Poission distributed [79], which is important for sampling from this distribution as detailed in Section 2.3.3.2.

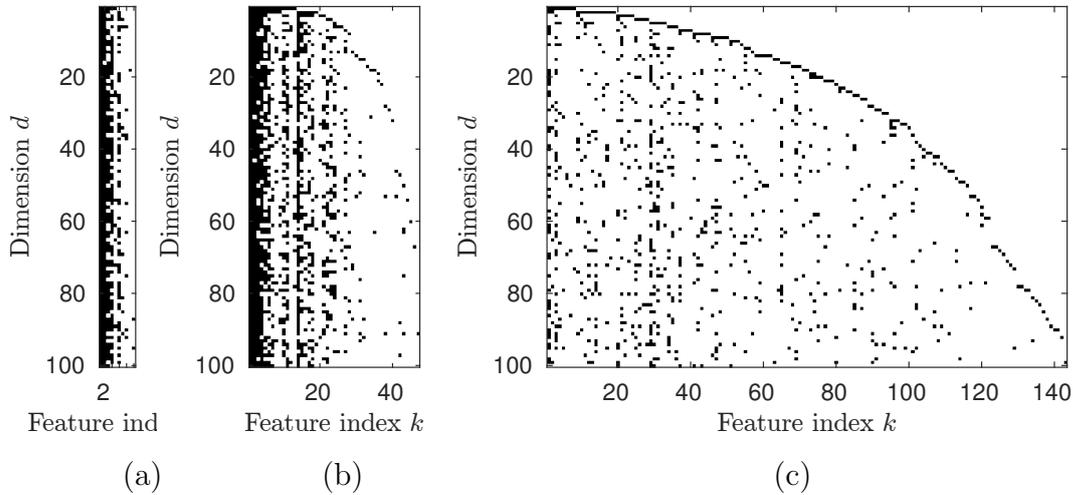


Figure 2.4. Different activation matrices obtained by sampling from the Indian Buffet Process with varying hyperparameter β_a and fixed $\alpha_a = 10$, (a) $\beta_a = 0.2$, (b) $\beta_a = 1$, and (c) $\beta_a = 5$. The sparsity is increased with β_a , resulting in more sampled features.

From the expression for the number of active features, \bar{K}_+ , it becomes apparent how the hyperparameters α_a and β_a influence the number of active features. The probability that an element in \mathbf{A}_∞ is active increases with both α_a and β_a . At the same time, the expected number of active features grows linearly with α_a , yielding sparse realizations. Consequently, α_a controls the number of active features. In contrast, β_a has only little effect on the number of active features. In particular, for $\beta_a \rightarrow \infty$, we obtain $\bar{K}_+ = \alpha_a D$, showing that the expected number of active features becomes independent of β_a for large β_a . Thus, β_a can be understood as controlling the density of the realizations. This is illustrated in Fig. 2.4.

Though the IBP models an infinite number of features, the number of active features is finite due to the sparsity assumption. Since we only need to store the active features in memory, the realizations are finite-sized matrices. Depending on the choice of the hyperparameters, the realizations can be sparse as well as dense.

2.3.3.2 Sampling from the Indian Buffet Process

Samples of feature representations are easily obtained by sampling from Eq. (2.11). For this, a Gibbs sampler [78] is used with a subsequent ordering of the activations. From the finite model in Eq. (2.10), the conditional for sampling the (k, d) th element

of \mathbf{A} , $a_{k,d}$ can be derived as [78]

$$P(a_{d,k} = 1 \mid \mathbf{a}_{k \setminus d}) = \frac{m_{k \setminus d} + \frac{\alpha_a \beta_a}{K}}{D + \frac{\alpha_a \beta_a}{K} + \beta_a - 1},$$

where $\mathbf{a}_{k \setminus d}$ is the k th column of \mathbf{A} without $a_{k,d}$ and $m_{k \setminus d}$ is the sum over the elements of $\mathbf{a}_{k \setminus d}$. Considering the limit for $K \rightarrow \infty$ results in [78]

$$P(a_{\infty,d,k} = 1 \mid \mathbf{a}_{\infty,k \setminus d}) = \frac{m_{\infty,k \setminus d}}{D + \beta_a - 1}.$$

Note that there is a certain probability that each element of the observation, indexed by d , $d = 1, \dots, D$, has been generated by a feature that has not been inferred yet. Assuming exchangeability, the ordering of the variables $a_{\infty,d,k}$ becomes irrelevant [15]. Thus, each element can be considered as the last being sampled such that the probability of activating K'_+ new features for the d th element is independent of its index [78],

$$P(K'_+ \mid -) \sim \text{Poisson}_{K^+} \left(\frac{\alpha_a \beta_a}{\beta_a + D - 1} \right), \quad (2.12)$$

where the bar symbol $(-)$ refers to all variables except K'_+ .

In summary, sampling \mathbf{A} works as follows. We set $a_{\infty,d,k}$ to one with probability $\frac{m_{\infty,k \setminus d}}{D + \beta_a - 1}$. With probability $P(K'_+ \mid -)$, we add K'_+ elements to the d th row of \mathbf{A} . After having iterated over all active rows, a proposal is made to remove all columns that contain zero entries only, resulting in a sample of a binary matrix with K_+ active rows, i.e., K_+ features.

Note that the samples generated by means of this algorithm need to be ordered if we want to sample feature class representations from the distribution in Eq. (2.11). The ordering can be achieved by means of the lof-operator described in [79] that sorts the columns of the transposed of \mathbf{A}_∞ into a left-ordered form (lof).

2.3.3.3 Sampling the Hyperparameters α_a and β_a

The hyperparameters α_a and β_a can be considered as Gamma distributed variables with their own hyperparameters, $h_{\alpha_A}^{(1)}, h_{\alpha_A}^{(2)}, h_{\beta_A}^{(1)}$ and $h_{\beta_A}^{(2)}$ [75], i.e.,

$$\begin{aligned} p(\alpha_a) &= \text{Ga}_{\alpha_a} \left(h_{\alpha_A}^{(1)}, h_{\alpha_A}^{(2)} \right), \\ p(\beta_a) &= \text{Ga}_{\beta_a} \left(h_{\beta_A}^{(1)}, h_{\beta_A}^{(2)} \right). \end{aligned}$$

The conditional of α_a is given as [75]

$$p(\alpha_a | -) = \text{Ga}_{\alpha_a} \left(K + h_{\alpha_A}^{(1)}, \sum_{d=1}^D \frac{\beta_a}{\beta_a + d - 1} + h_{\alpha_A}^{(2)} \right),$$

which can be sampled from efficiently. Since the likelihood and the prior of β_a are not conjugate, we use a Metropolis step with the prior of β_a as proposal distribution. The acceptance ratio r_{β_a} is then given as [75]

$$r_{\beta_a} = \frac{p(\beta_a' | -)}{p(\beta_a | -)} = \frac{P(\mathbf{A} | \alpha_a, \beta_a')}{P(\mathbf{A} | \alpha_a, \beta_a)},$$

where β_a' denotes the proposed value.

2.4 Clustering and Classification

In this section, we revisit different clustering and classification methods that are utilized in Chapter 3. First, spectral clustering is briefly explained which we consider in the proposed band selection algorithm in Section 3.3. Second, the k -Nearest Neighbor (KNN) and the SVM classifier are introduced. Both classifiers are used for evaluation throughout Chapter 3.

2.4.1 Spectral Clustering

Many clustering algorithms such as k -means [80,81] or Expectation Maximization (EM) [82] for Gaussian mixture models pose strong assumptions on the distribution of the observations, which in practice may not hold and therefore lead to poor results. An alternative is the graph cut algorithm that has demonstrated accurate clustering results in many image processing problems [83]. For image segmentation, the graph cut assumes a graph-based representation of the image, where the nodes are the elements of the image and the edges represent the affinities between the elements (high values indicate a high similarity between nodes). A sink and a source need to be defined, which represent pixels belonging to separate segments. According to the max-flow min-cut theorem [84], cutting the graph along the maximum flow between sink and source minimizes the energy and yields the segmentation result [83].

However, graph cut suffers from the problem that the sizes of the resulting clusters can be heavily unbalanced. Normalized cuts [85] overcome this problem by normalizing the

cut by the volume of the segments. However, minimizing normalized cuts is an NP-complete problem [85]. A relaxation of normalized cuts is given by Spectral Clustering (SC) [86].

In the following, we briefly describe self-tuning SC [86, 87]. Let $\mathbf{z}_n \in \mathbb{R}^{D \times 1}$, with $n = 1, \dots, N_z$, denote N_z D -dimensional observations that shall be clustered. For this, the normalized graph Laplacian, \mathbf{L}_{SC} , is defined [86],

$$\mathbf{L}_{\text{SC}} = \mathbf{D}_{\text{SC}}^{-1/2} \mathbf{A}_{\text{SC}} \mathbf{D}_{\text{SC}}^{-1/2},$$

where \mathbf{A}_{SC} is an $N_z \times N_z$ matrix that describes the affinities between all features. The affinity between the n th and m th observation is defined as

$$a_{\text{SC},n,m} = \exp\left\{-\frac{1}{\gamma_{\text{SC}}^2} \|\mathbf{z}_n - \mathbf{z}_m\|_2^2\right\} \quad \text{for } n \neq m, \quad (2.13)$$

with γ_{SC} denoting a scaling factor. The self-affinities, $a_{\text{SC},n,n}$, $n = 1, \dots, N_z$, are set to zero. The values of the diagonal matrix \mathbf{D}_{SC} are computed as the sums over the rows of \mathbf{A}_{SC} . Due to the nonlinearity in Eq. (2.13), scaling the observed data is crucial for successful clustering. We provide more details on this issue in Section 2.4.3.2.

Ideally, the number of clusters of the observations is given by the number of eigenvalues of the graph Laplacian that are equal to one [88]. In practice, due to noise and modeling errors, many eigenvalues may take values close to one, making a clear decision difficult. Thus, one resorts to setting K_{SC} to the number of eigenvalues exceeding a predefined threshold. Stacking the corresponding K_{SC} eigenvectors yields the transformed observations $\mathbf{V}_{\text{SC}} \in \mathbb{R}^{N_z \times K_{\text{SC}}}$. A conventional clustering algorithm, such as k -means, can then be used to cluster the transformed observations contained in the rows of \mathbf{V}_{SC} .

The value of the scaling factor, γ_{SC} , and the number of final clusters, K_{SC} , have a strong impact on the result. Perona and Zelnik-Manor [87] present an efficient method to estimate local scales and a suitable number of clusters. A local scale, $\gamma_{\text{SC},n}$, $n = 1, \dots, N_z$, can be considered as the distance between \mathbf{z}_n and its k th nearest neighbor where k is determined by the dimension of the observation vector [87]. The affinity in Eq. (2.13) is then reformulated as

$$\tilde{a}_{\text{SC},n,m} = \exp\left\{-\frac{1}{\gamma_{\text{SC},n} \gamma_{\text{SC},m}} \|\mathbf{z}_n - \mathbf{z}_m\|_2^2\right\},$$

with $n = 1, \dots, N_z$ and $m = 1, \dots, N_z$. In order to estimate the number of clusters, K , Perona and Zelnik-Manor [87] propose a cost function that aims at aligning \mathbf{V}_{SC} to the canonical coordinate system. Minimizing this cost function with respect to a rotation matrix implicitly yields the number of clusters. For details, we refer the reader to [87, 89, 90].

2.4.2 k -Nearest Neighbor

The k -Nearest Neighbor (KNN) classifier [91] is a simple, yet effective classification method. The algorithm can be derived as follows. The probability that the observation $\mathbf{z} \in \mathcal{Z}^{D \times 1}$ is located in region \mathcal{R} is equal to the distribution of \mathbf{z} integrated over \mathcal{R} ,

$$P(\mathbf{z} \in \mathcal{R}) = \int_{\mathcal{R}} p(\mathbf{z}') d\mathbf{z}'. \quad (2.14)$$

Assuming that \mathcal{R} is sufficiently small, the distribution over \mathbf{z} , $p(\mathbf{z})$, is constant in this region. Thus, the integral can be written as

$$\int_{\mathcal{R}} p(\mathbf{z}') d\mathbf{z}' = V_{\mathcal{R}} p(\mathbf{z}), \quad (2.15)$$

with $V_{\mathcal{R}}$ denoting the volume of \mathcal{R} . However, assuming \mathcal{R} is large and contains many training samples, the probability in Eq. (2.14) can be approximated as

$$P(\mathbf{z} \in \mathcal{R}) = \frac{K_{\mathcal{R}}}{N_{\mathbf{T}}}, \quad (2.16)$$

with $K_{\mathcal{R}}$ denoting the number of training samples in region \mathcal{R} and $N_{\mathbf{T}}$ the total number of training samples. Using Eq. (2.15) and Eq. (2.16), and substituting the expressions in Eq. (2.14), yields

$$p(\mathbf{z}) = \frac{K_{\mathcal{R}}}{V_{\mathcal{R}} N_{\mathbf{T}}}. \quad (2.17)$$

Note that for the KNN classifier, $K_{\mathcal{R}}$ is considered as a parameter and needs to be set *a priori*. If we condition only on the training samples belonging to the i th class, \mathcal{C}_i , with $i \in \{1, \dots, N_{\mathcal{C}}\}$, we obtain

$$p(\mathbf{z} | \mathcal{C}_i) = \frac{K_{\mathcal{R},i}}{N_{\mathbf{T},i} V_{\mathcal{R}}}, \quad (2.18)$$

where $K_{\mathcal{R},i}$ denotes the number of training samples of class \mathcal{C}_i in \mathcal{R} and $N_{\mathbf{T},i}$ the total number of training samples that are assigned to class \mathcal{C}_i . The prior for each class is approximated by the relative frequency,

$$P(\mathcal{C}_i) = \frac{N_{\mathbf{T},i}}{N_{\mathbf{T}}}. \quad (2.19)$$

Using Bayes' theorem, the posterior is expressed in terms of the evidence, class likelihood, and the prior, derived in Eqs. 2.17, 2.18, and 2.19,

$$\begin{aligned} P(\mathcal{C}_i | \mathbf{z}) &= \frac{p(\mathbf{z} | \mathcal{C}_i) P(\mathcal{C}_i)}{p(\mathbf{z})} \\ &= \frac{K_{\mathcal{R},i}}{K_{\mathcal{R}}}. \end{aligned}$$

Thus, classification using KNN means simply counting the number of training samples for the i th, $i = 1, \dots, N_C$, class in the $K_{\mathcal{R}}$ nearest neighborhood of \mathbf{z} . The observation \mathbf{z} is then assigned to the class with the highest probability, i.e., the highest count. The distance measure that determines the $K_{\mathcal{R}}$ nearest neighbors has a significant impact on the prediction result. Throughout the thesis, we use the Euclidean distance.

2.4.3 Support Vector Machine

The Support Vector Machine (SVM) is a discriminative, linear, binary classifier, which is founded on statistical learning theory [51, 54, 92]. The distance of the D -dimensional observation $\mathbf{z} \in \mathcal{Z}^{D \times 1}$ to the hyperplane that separates the two classes is proportional to

$$f_{\text{SVM}}(\mathbf{z}) = \mathbf{w}_{\text{SVM}}^T \mathbf{z} + b_{\text{SVM}},$$

with \mathbf{w}_{SVM} and b_{SVM} denoting the hyperplane parameters that need to be estimated from training data. The sign $y \in \{-1, 1\}$ of $f_{\text{SVM}}(\mathbf{z})$ indicates the class of \mathbf{z} informing on which side of the hyperplane the observation is located.

The key idea of the SVM is to maximize the margin between the classes as illustrated in Fig. 2.5. Consequently, the uncertainty about the classification, which is reflected by the region between the classes, is equally shared between the classes. This is based on the assumption that the cost of misclassifying an observation is equal for both classes. As the margin is proportional to $\frac{1}{\|\mathbf{w}_{\text{SVM}}\|_2^2}$, the SVM can be formulated as an optimization problem given as [51, 92]

$$\begin{aligned} \{\hat{\mathbf{w}}_{\text{SVM}}, \hat{b}_{\text{SVM}}\} &= \arg \min_{\mathbf{w}_{\text{SVM}}, b_{\text{SVM}}} \frac{1}{2} \|\mathbf{w}_{\text{SVM}}\|_2^2 \\ \text{s.t. } & y_n (\mathbf{w}_{\text{SVM}}^T \mathbf{z}_n + b_{\text{SVM}}) \geq 1 \quad \forall n = 1, \dots, N_T. \end{aligned} \quad (2.20)$$

The constraint ensures that all training data points are correctly classified and reside outside the margin. Hence, the SVM in this original form is only able to linearly separate two non-overlapping classes.

2.4.3.1 Nonlinear Separation and the Kernel Trick

For nonlinear separation, the observations are transformed [54, 92] by a nonlinear function $\phi: \mathcal{Z}^{D \times 1} \rightarrow \mathcal{F}$, with \mathcal{F} denoting a feature space. The transform is chosen such that the observation is mapped into a space of arbitrary dimension where the data can

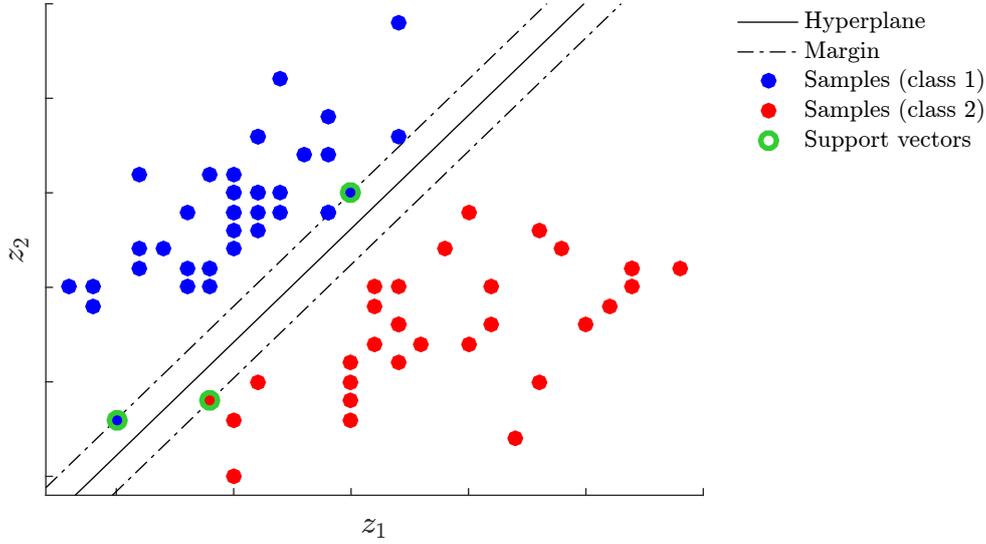


Figure 2.5. Illustration of two-dimensional observations that are separated by the hyperplane of the SVM. The support vectors defining the hyperplane are highlighted in green color.

be easily separated. Replacing \mathbf{z} by $\phi(\mathbf{z})$ and rewriting the optimization problem in Eq. (2.20) as a Lagrangian with multipliers $\alpha_{\text{SVM},n}$, $n = 1, \dots, N_{\text{T}}$, yields

$$\mathbf{L}(\mathbf{w}_{\text{SVM}}, \boldsymbol{\alpha}_{\text{SVM}}) = \frac{1}{2} \|\mathbf{w}_{\text{SVM}}\|_2^2 - \sum_{n=1}^{N_{\text{T}}} \alpha_{\text{SVM},n} (y_n (\mathbf{w}_{\text{SVM}}^{\text{T}} \phi(\mathbf{z}_n) + b_{\text{SVM}}) - 1). \quad (2.21)$$

The maximum of $\mathbf{L}(\mathbf{w}_{\text{SVM}}, \boldsymbol{\alpha}_{\text{SVM}})$ with respect to \mathbf{w}_{SVM} and b_{SVM} can be found by setting the derivatives to zero [54],

$$\mathbf{w}_{\text{SVM}} = \sum_{n=1}^{N_{\text{T}}} \alpha_{\text{SVM},n} y_n \phi(\mathbf{z}_n), \quad (2.22)$$

$$0 = \sum_{n=1}^{N_{\text{T}}} \alpha_{\text{SVM},n} y_n. \quad (2.23)$$

Replacing \mathbf{w}_{SVM} in Eq. (2.21) by the expression in Eq. (2.22), we obtain the *Wolfe dual representation* [93]. The Lagrangian can then be rewritten as

$$\mathbf{L}_{\text{Dual}}(\boldsymbol{\alpha}_{\text{SVM}}) = \sum_{n=1}^{N_{\text{T}}} \alpha_{\text{SVM},n} - \frac{1}{2} \sum_{n=1}^{N_{\text{T}}} \sum_{m=1}^{N_{\text{T}}} \alpha_{\text{SVM},n} \alpha_{\text{SVM},m} y_n y_m \underbrace{\phi(\mathbf{z}_n)^{\text{T}} \phi(\mathbf{z}_m)}_{=K_{\text{SVM}}(\mathbf{z}_n, \mathbf{z}_m)}.$$

In the *dual representation*, the transform $\phi(\mathbf{z})$ appears as an inner product with itself. Hence, instead of computing the transform and the inner product separately,

the outcome can be computed directly by means of the kernel function $K_{\text{SVM}}(\mathbf{z}, \mathbf{z}') = \phi(\mathbf{z})^T \phi(\mathbf{z}')$. Substituting the inner products of $\phi(\mathbf{z})$ by $K_{\text{SVM}}(\mathbf{z}, \mathbf{z}')$ is referred to as the *kernel trick*. The kernel function can be regarded as a similarity measure with high values of the kernel function indicating strong similarity between the observations $\mathbf{z} \in \mathcal{Z}^{D \times 1}$ and $\mathbf{z}' \in \mathcal{Z}^{D \times 1}$.

The use of the kernel function brings several benefits. Instead of defining a feature mapping, we can design the kernel and compute the inner product directly, without the expensive need of computing the product in a high-dimensional space. Basically any function can be used as a kernel function, if, for any possible inputs \mathbf{z} and \mathbf{z}' , the function is positive semidefinite [94]. Though this is, generally, not trivial to show, rules have been derived to construct new kernels given a set of valid kernels [54].

During training, many of the Lagrangian multipliers $\alpha_{\text{SVM},n}$, $n = 1 \dots N_T$, will become zero since the margin is sufficiently defined by only few support vectors as demonstrated in Fig. 2.5. For this reason, the SVM is a sparse learning method.

2.4.3.2 Radial Basis Function

Besides faster computation, the kernel trick allows to transform the data into an infinite dimensional feature space, where the classes can be always separated. This is achieved by the Gaussian kernel, also referred to as Radial Basis Function (RBF) [54], which is defined as

$$K_{\text{RBF}}(\mathbf{z}, \mathbf{z}') = \exp\left(-\frac{1}{\gamma_{\text{RBF}}^2} \|\mathbf{z} - \mathbf{z}'\|_2^2\right).$$

Rewriting the exponential function using Taylor series expansion results in a polynomial of infinite degree, which explains the mapping into an infinite dimensional feature space. The parameter γ_{RBF} defines the bandwidth of the kernel and is usually adapted via cross-validation [95]. As rule of thumb, this parameter can be initialized by the mean or median of the distances between all observations.

Note that a normalization of the observations is required when utilizing the RBF kernel to improve the separability of the classes [96]. A common way is to rescale the elements z_d , $d = 1, \dots, D$, of the observation to the range of $[-1, 1]$. An explanation for this requirement can be found in the shape of the squared exponential function illustrated in Fig. 2.6. For low values, small differences of the input variables, denoted by I_1 , lead to highly different values of the kernel, denoted by K_h , and thus good separability. If the differences reside in an upper region I_h , the outcome of the function will differ less, as indicated by K_1 , which makes separation difficult. Of course, numerical stability is another reason [96].

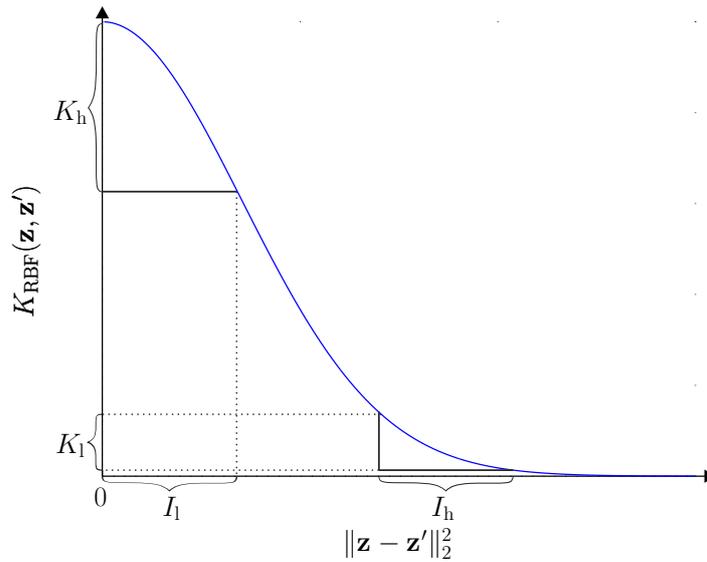


Figure 2.6. Illustration of the importance of scaling the input data when utilizing a RBF kernel. For large input values, the difference of the function output is rather small compared with the output of small input values.

2.4.3.3 Overlapping Classes

Often, the observations cannot be as easily separated as in Fig. 2.5. In fact, the classes often overlap making even nonlinear separation difficult. Using an RBF kernel with parameters fitted optimally to the training data, it is possible to achieve little training errors even for difficult data sets. This, however, leads to strong overfitting and poor prediction performance for new observations.

A solution consists in accepting a violation of the margin constraint in Eq. (2.20) for few samples, which may then reside within the margin. Consequently, not all training data is correctly classified, but a higher generalization of the classifier can be achieved. The violation is introduced by a *slack variable*, $\xi_n \in \mathbb{R}_0^+$, $n = 1, \dots, N_T$, for each training sample. The constraint optimization problem is then changed to

$$\begin{aligned} \{\hat{\mathbf{w}}_{\text{SVM}}, \hat{b}_{\text{SVM}}\} &= \arg \min_{\mathbf{w}_{\text{SVM}}, b_{\text{SVM}}, \boldsymbol{\xi}} \frac{1}{2} \|\mathbf{w}_{\text{SVM}}\|_2^2 + C_{\text{SVM}} \sum_{n=1}^{N_T} \xi_n \\ \text{s.t. } & y_n (\mathbf{w}_{\text{SVM}}^T \mathbf{z}_n + b_{\text{SVM}}) \geq 1 - \xi_n \quad \forall n = 1 \dots N_T. \end{aligned} \quad (2.24)$$

Eq. (2.24) shows that the slack variables, and, hence, the tolerated violation of the margin, are upper bounded with the bound depending on C_{SVM} . If $C_{\text{SVM}} \rightarrow \infty$, violations are strictly prohibited, which leads to the standard formulation in Eq. (2.20). The parameter C_{SVM} can also be learned by means of cross-validation.

Note that the results of the Lagrangian multipliers and the kernel trick are still valid, where the necessary adaptations are straightforward. Further details can be found in [51, 54].

2.4.3.4 Extension to Multiclass-Classification

A crucial limitation of the SVM is the restriction to binary classification. There exist basically two approaches to enable multiclass-classification with SVMs: *one-versus-rest* and *one-versus-one* [97]. In one-versus-rest classification, we train for each class an SVM. As the name suggests, the first class is one of the N_C classes while the second class comprises all $N_C - 1$ remaining classes. For prediction, the observed sample is assigned to the class showing the largest distance to the margin. A key problem is the assumption that the SVMs are equally scaled, which is required for a valid distance-based comparison. Since the SVMs are trained independently, this cannot be guaranteed [54]. Further, the *one-versus-rest* scheme may suffer from imbalances training samples [54].

In the one-versus-one scheme, $\frac{N_C}{2} (N_C - 1)$ SVMs are trained, meaning that each two-class combination is considered. During prediction, an observation is assigned to the class it has been most often assigned to. This strategy is referred to as majority voting. Using this scheme, we overcome the problem of comparing distances but, therefore, the number of classifiers to be trained grows quadratically with the number of classes instead of linearly as in the one-versus-rest scheme.

Though multiclass formulations of SVMs have been derived, e.g., in [98], there is no theoretical result on which method performs best. In [99] the authors empirically show that the one-vs-one scheme provides slightly better results than one-vs-rest. For this reason, we use one-vs-one classification throughout the thesis.

Chapter 3

Feature Learning for Classification in Hyperspectral Imaging

In this chapter, we investigate low-dimensional representations for classification in Hyperspectral Imaging (HSI).

In the first part of this chapter, we investigate a feature selection algorithm for the classification of hyperspectral images. This approach is based on clustering similar bands, hereby reducing the dimension of the feature vector significantly. In many cases this yields higher classification accuracies than state-of-the-art approaches. In the second part, we consider the problem of acquiring a low-dimensional representation of the image directly. As in classification we are not interested in the image itself, we skip the step of reconstructing the image and perform classification directly on the low-dimensional observations.

In Section 3.1, we start with an introduction and motivation for feature learning in HSI. Before explaining the proposed approaches, the data sets used for evaluation in both models are introduced in Section 3.2. A feature selection approach, referred to as band selection in HSI, is then presented in Section 3.3. A more efficient approach in comparison to traditional sensing in terms of data acquisition is explained in Section 3.4, where CS is utilized to directly capture images in a low-dimensional representation.¹

3.1 Motivation and Introduction

HSI is an emerging optical sensing technique that allows to simultaneously capture hundreds of bands of the visible and infrared light range. Since each material possesses a characteristic spectrum, referred to as signature, scene analysis and classification based on HSI becomes fairly easy [102]. Today, HSI is commonly used in remote sensing, e.g., in agriculture, urban mapping, and security applications [1]. Thanks to efficient algorithms and advanced hardware, HSI also finds application in other fields, such as, e.g., in nutrition analysis [103] and waste management [104].

Due to the high number of bands, processing hyperspectral images is often computationally challenging. Besides high storage demands, post-processing and data analysis

¹Parts of the work in this chapter were presented at EUSIPCO 2013 [100] and ICASSP 2014 [101]

are often time-consuming tasks, requiring costly computation resources. Especially when we are interested in classifying the image, high-dimensional observations can lead to poor results due to the curse of dimensionality (Section 2.1.3). Therefore, a key challenge is to reduce the dimension of the observations by means of feature extraction. Feature selection or learning can be both performed in the spatial as well spectral domain of the hyperspectral image.

Spatial information can be exploited by extracting information from neighboring pixels. For example, in [105] a watershed transform is applied to the hyperspectral image, leading to a segmentation map that is incorporated into the classification process. The authors of [106, 107] propose the use of morphological features combined with the spectral information of each pixel to improve classification accuracy. In context of the classification challenge of the *2013 IEEE Geoscience and Remote Sensing Data Fusion Contest*, we designed features for specific tasks, e.g., for the detection of roads, buildings, and parking lots. Combined with ensemble learning, we obtained outstanding results, resulting in the highest classification accuracy of all submitted approaches [108]. Recent approaches for automatically extracting spatial features are largely based on Deep Learning (DL), c.f. [109–112].

In the following, we consider pixel-wise classification and present algorithms that aim at reducing the number of bands for classification. As the bands serve as features for the classification, we use the terms *bands* and *features* interchangeably in this chapter. For this purpose, several methods have been proposed to date, e.g., based on Contrast Measure and Correlation Measure [113] as well as Principal Component Analysis (PCA) [114]. In [114], a modification of PCA is presented, where the basis vectors are chosen according to the Signal-to-Noise Ratio (SNR) instead of the variance to reduce the effect of noise. These methods basically have two disadvantages: first, they heavily depend on suitable parameter selection and second, the implied transform of the observations may lead to information loss and, hence, low classification accuracy [115]. For these reasons, in [115], different Constrained Band Selection (CBS) methods are presented. The key idea is to constrain each band of the hyperspectral image, while minimizing the correlation with the other bands. In Linearly Constrained Minimum Variance (LCMV), the bands are constrained based on one out of four proposed correlation-based criteria [115], where the number of bands that should be retained is estimated by means of Virtual Dimensionality (VD) [116]. In [117] and [118], the authors propose the use of clustering algorithms for band selection. In many experiments, clustering-based approaches have outperformed previous methods in terms of classification accuracy. Martinez-Uso *et al.* [117] cluster the bands by similarity measures such as the Kullback-Leibler divergence, while Qian *et al.* [118] utilize Affinity Propagation (AP). Further, Qian *et al.* utilize wavelet shrinkage to remove noisy bands. For these reasons, we will

investigate a spectral clustering-based band selection algorithm in the first part of this chapter.

Following the classic band selection scheme, data is first costly acquired and then literally thrown away during band selection. A more efficient method is to capture only relevant information about the image. This can be achieved by CS, which allows to acquire a signal of interest by sampling below the Nyquist-Shannon rate [45, 46] as explained in Section 2.3.2. The CS framework has been successfully applied to HSI, e.g., in [119–121], resulting in a low-dimensional acquisition of the hyperspectral data. In the second part of this chapter, we extend our approach proposed in [121] to a Compressive Classification (CC) framework, as ultimately, in many applications, we are not interested in the hyperspectral image itself, but in the content which can be extracted by means of classification. For this, we build on [122] to introduce a CC framework for HSI that allows for the acquisition of the pixels in the compressed domain, in which they are directly classified. The advantage of this method is that a costly reconstruction of the hyperspectral image is not required, remarkably reducing computation costs. Further, errors due to poor reconstructions are avoided.

3.2 Hyperspectral Image Data Sets

The proposed methods are evaluated based on three different data sets: (i) Indiana’s Indian Pines (IP), (ii) University of Pavia (UP), and (iii) Center of Pavia (CP). The IP data set shows mainly vegetation at a low spatial resolution. In contrast, the UP and CP data sets contain urban area at high spatial resolutions. Details about the data sets are given in the following.

3.2.1 Indiana’s Indian Pines (IP)

The IP image was captured by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) during a flight a few miles west of Lafayette, Indiana, USA in June, 1992. This scene is often used in HSI research since it is publicly available [123].

The size of the image is 145×145 pixels, showing mainly vegetation at a resolution of $20 \frac{\text{m}}{\text{pixel}}$. After removing several water absorption bands, the image contains 200 bands [124]. Fig. 3.1 shows the real color image of the scene. The ground truth mapping reveals that basically 16 different classes occur in the scene, while most of

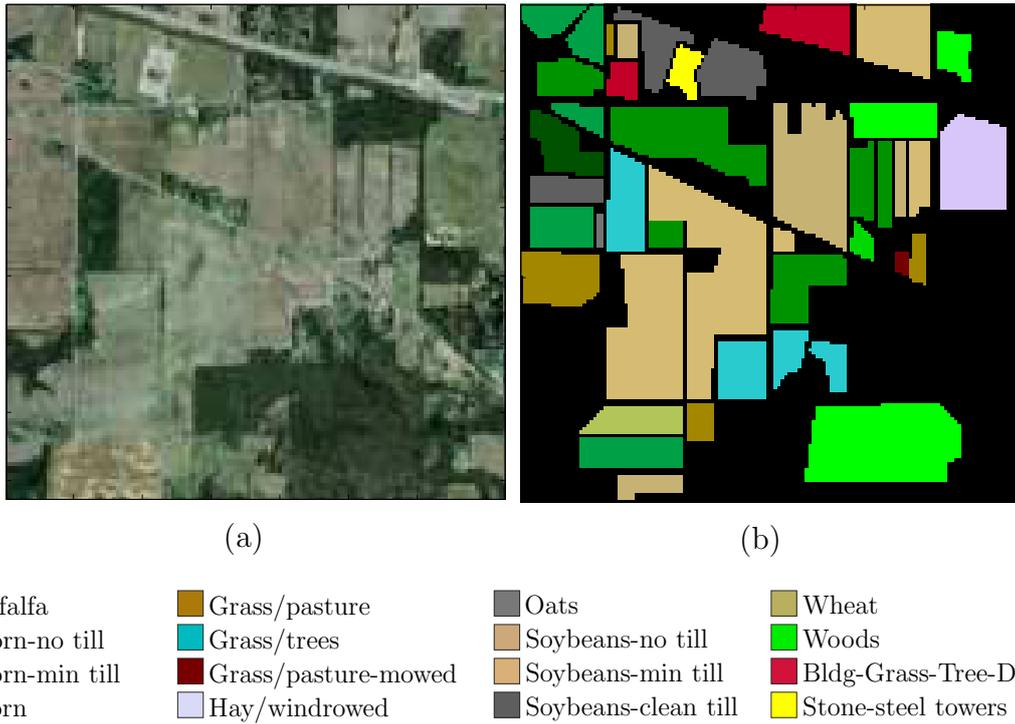


Figure 3.1. (a) True color image of the *Indian Pines* data set and (b) the ground truth, containing 16 different classes of mainly vegetation.

them represent vegetation. Due to the low spatial resolution, it is likely that a pixel represents not only one, but rather a mixture of different materials. This issue can be alleviated by unmixing the pixels, which is detailed in Chapter 4.

3.2.2 University of Pavia (UP) and Center of Pavia (CP)

The UP and CP images were taken by the Digital Airborne Imaging Spectrometer (DAIS) and Reflective Optics System Imaging Spectrometer (ROSIS) during a flight conducted by *Deutsche Zentrum für Luft- und Raumfahrt* (DLR) in 2002. The systems produced images of 610×340 and 1096×490 pixels, respectively with 115 bands. For the UP data set, 12 noisy bands were removed, resulting in 103 bands for further processing. In the CP image, 13 bands were removed, leaving 102 bands. During the flight, ground and air measurements were taken, allowing for atmospheric correction of the images.

The spatial resolution of both images is considerably higher with $1.3 \frac{\text{m}}{\text{pixel}}$ (UP) and $2.6 \frac{\text{m}}{\text{pixel}}$ (CP) compared with the IP image. Since the Pavia images shows an urban

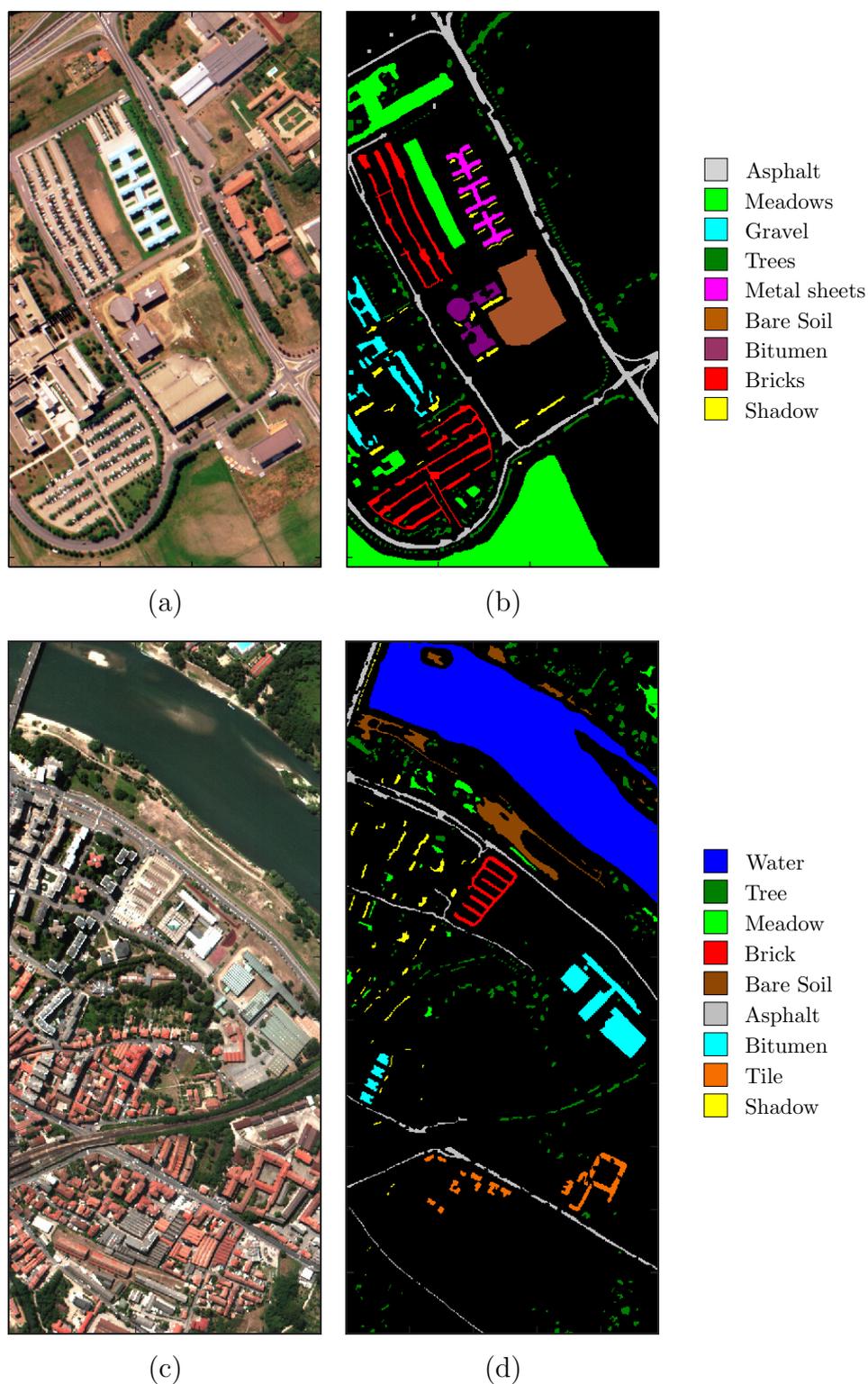


Figure 3.2. (a) True color image of the *University of Pavia* data set and (b) the ground truth, as well as the (c) true color image of the *Center of Pavia* data set and (d) the ground truth. Both image contain 9 classes of different materials.

area, the 9 classes in the scenes present distinct materials, such as metal, vegetation, or asphalt. The UP and CP images are depicted in Fig. 3.2.

3.2.3 Acknowledgment

We would like to thank Prof. David Landgrebe, Purdue University, USA, for kindly providing the Indiana’s Indian Pines data set and Prof. Paolo Gamba, University of Pavia, Italia, for providing the University of Pavia and the Center of Pavia images [125].

3.3 Band Selection by Means of Spectral Clustering

In this section, we propose a clustering-based band selection algorithm that groups similar bands of the hyperspectral image into clusters. Representatives, which are extracted from each cluster, are finally considered for classification. This method provides high classification accuracy with the tested classifiers. A remarkable advantage of the proposed algorithm compared with many other approaches is that the parameters are automatically adapted and, thus, need not to be set by an expert.

The key idea of this approach is to group similar bands first. Ideally, each group, or cluster, can then be represented by a single feature. However, as a single feature may not be able to capture the relevant information of a cluster, we suggest to select multiple representatives such that the information content of the cluster is retained at a pre-fixed rate. Thus, the proposed band selection scheme is composed of two parts: (i) cluster formation and (ii) representative selection. In the first step, the algorithm groups similar bands into clusters. In the second step, features are extracted from each cluster. The extracted features, representing the image, can be further processed, e.g., for classification. An overview of this procedure is depicted in Fig. 3.3.²

3.3.1 Cluster Formation

Let $\mathbf{Z}_{\text{HSI}} \in \mathbb{R}^{N_1 \times N_2 \times D}$ denote the observed hyperspectral image of $N_z = N_1 \times N_2$ pixels and D bands. We consider the vectorized bands $\mathbf{b}_d \in \mathbb{R}^{N_z}$, with $d = 1, \dots, D$, as feature vectors for the clustering algorithm. For clustering, basically any method can

²Parts of the work in this section were presented at EUSIPCO 2013 [100].

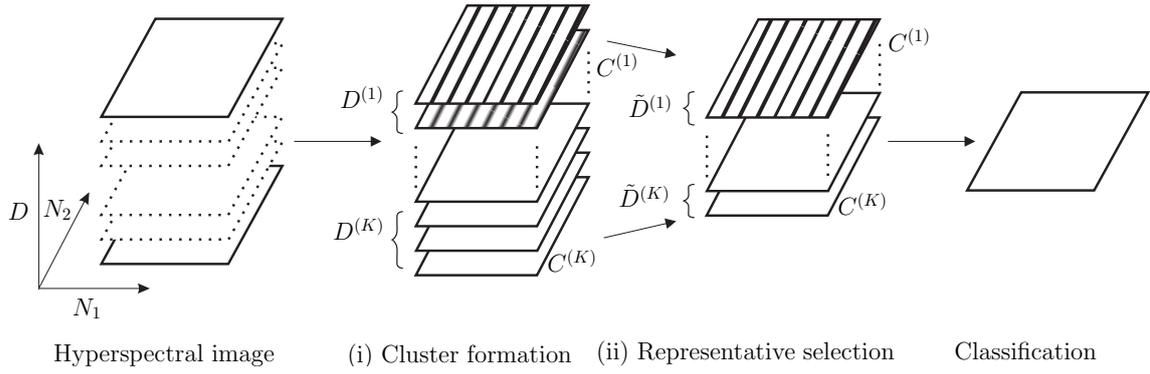


Figure 3.3. Overview of the proposed band selection approach. First, the hyperspectral bands are clustered. Second, from each cluster, representatives are selected which serve as features for classification.

be used, e.g., k -means. We propose the use of self-tuning spectral clustering [87] as this method is able to automatically adjust the required parameters. In contrast to many other methods, e.g., k -means, spectral clustering does not make any assumptions on the distribution of the feature vectors, which helps to avoid modeling errors. After clustering, the bands are grouped into K_{SC} clusters $C^{(k)}$, $k = 1, \dots, K_{\text{SC}}$. In the subsequent step, representatives of each cluster are extracted.

3.3.2 Representative Selection

For each of the K_{SC} clusters, low-dimensional representations have to be found. The representations can be obtained by means of a mapping $\phi_{\text{SC}} : \mathbb{R}^{D^{(k)}} \rightarrow \mathbb{R}^{\tilde{D}^{(k)}}$ where $D^{(k)}$ is the number of bands and $\tilde{D}^{(k)} < D^{(k)}$ is the number of reduced bands in the k th cluster. For this purpose, we have tested many different methods, e.g., Contrast Measure (CM), pooling, and VD [113]. In our experiments, PCA has shown the best performance. This is likely due to the fact that representatives chosen by PCA capture the variation of the data. As the variation reflects the amount of information, PCA can be understood as extracting the most informative representatives.

Thus, PCA is applied to each cluster to obtain the representations. First, the covariance matrix $\Sigma^{(k)} \in \mathbb{R}^{D^{(k)} \times D^{(k)}}$ of the bands is estimated for each cluster $C^{(k)}$. Second, the eigenvalues, $\lambda_d^{(k)}$, and the corresponding eigenvectors, $\mathbf{v}_d^{(k)}$, with $d = 1, \dots, D^{(k)}$, of $\Sigma^{(k)}$ are computed. The eigenbasis of each cluster is then given by

$$\mathbf{V}^{(k)} = \left[\mathbf{v}_1^{(k)}, \dots, \mathbf{v}_{D^{(k)}}^{(k)} \right]. \quad (3.1)$$

Note that we assume that the eigenvectors and eigenvalues are sorted in descending order of the magnitude of the eigenvalues.

Third, each cluster is represented by the eigenvector showing the largest eigenvalue only. As the number of clusters, K_{SC} , inferred by spectral clustering can be low, representing each cluster by only one feature, i.e., $\tilde{D}^{(k)} = 1$, can lead to a strong reduction of the feature space. If the dimension of the feature space becomes too low, discriminating the classes will be difficult, resulting in a decreased classification accuracy. For this reason, each cluster is represented by $\tilde{D}^{(k)}$ features, where $\tilde{D}^{(k)}$ is estimated by means of the information content, $\text{IC}^{(k)}$, of the k th cluster. We define the information content as the sum over all eigenvalues,

$$\text{IC}^{(k)} = \sum_{d=1}^{D^{(k)}} \lambda_d^{(k)}, \quad (3.2)$$

which reflects the variation of data in the k th cluster [113]. As we aim at reducing the number of eigenvectors while maintaining the information content, we suggest to determine the number of kept eigenvalues $\tilde{D}^{(k)}$ for each cluster by finding a minimal $\tilde{D}^{(k)}$ that fulfills

$$\sum_{d=1}^{\tilde{D}^{(k)}} \lambda_d^{(k)} \geq \alpha_{\text{IC}} \text{IC}^{(k)}, \quad (3.3)$$

where α_{IC} , $0 \leq \alpha_{\text{IC}} \leq 1$, is the information content parameter that controls the fraction of content to be kept. By stacking the eigenvectors having the $\tilde{D}^{(k)}$ largest eigenvalues, we finally obtain the reduced eigenbasis

$$\tilde{\mathbf{V}}^{(k)} = \left[\mathbf{v}_1^{(k)}, \dots, \mathbf{v}_{\tilde{D}^{(k)}}^{(k)} \right].$$

Consequently, the presented scheme implicitly estimates the number of bands, which is equal to the number of features, in dependence of α_{IC} . Thus, the total number of features, K , is given as the number of representatives of all clusters,

$$K = \sum_{k=1}^{K_{\text{SC}}} \tilde{D}^{(k)}. \quad (3.4)$$

3.3.3 Classification

During training, the basis matrices, $\tilde{\mathbf{V}}^{(k)}$, of each of the clusters $C^{(k)}$, $k = 1, \dots, K_{\text{SC}}$, are learned. With these transforms, the observations are projected onto a lower dimensional space,

$$\mathbf{s}_n = \left[\tilde{\mathbf{V}}^{(1)} \quad \dots \quad \tilde{\mathbf{V}}^{(K_{\text{SC}})} \right]^T \mathbf{z}_n,$$

where $\mathbf{z}_n \in \mathbb{R}^{D \times 1}$ is the n th pixel of the image with $n = 1, \dots, N_z$ and $\mathbf{s}_n \in \mathbb{R}^{K \times 1}$ its projection. Finally, we obtain the low-dimensional representations of the observations,

$\mathbf{S} \in \mathbb{R}^{K \times N_z}$, by stacking the projected observations. As each projection represents one pixel, the projections can be classified directly resulting in a pixel-wise approach. Alternatively, spatial filtering or spatial feature extraction, such as morphological operations [126], can be applied to improve the classification performance by exploiting information provided by neighboring pixels.

3.3.4 Results

In this section, the proposed method, Spectral Clustering-Based Band Selection (SCBS), is compared with state-of-the-art band selection methods with respect to classification accuracy. For this purpose, we evaluate the Overall Accuracy (OA), which is the fraction of all correctly classified samples versus all samples and the Class Accuracy (CA) reflecting the number of correctly classified samples of each class. In the following, we present results for the IP, CP [125] as well as the UP [125] data sets.

For classification, we consider the KNN [91] classifier and SVMs [127]. The experiments have been repeated ten times where the training samples have been randomly chosen. For IP, we consider in total 660 training samples, for CP 5536 and UP 3921. By means of cross-validation [95], the parameters of the classifiers have been estimated, i.e., the number of neighbors $K_{\mathcal{R}}$ for KNN and the regularization C_{SVM} as well as the kernel bandwidth γ_{SVM} for the SVMs. First, we investigate the effect of different numbers of bands on the classification accuracy. Second, we use the information content approach to estimate a suitable number of bands automatically.

3.3.4.1 Influence of the Number of Bands

In this section, we investigate the effect of the number of bands for the classification. Therefore, instead of estimating the number of features by means of the information content, we set K manually between 5 and 50 with a step size of 5. In order to compare the proposed method, SCBS, with recent methods, we also show results for AP [118], LCMV [115], Maximum-Variance PCA (MVPCA) [114], and all bands (All) using the same setup.

Using KNN for the IP image, the proposed method performs similar to AP leading to significantly higher accuracies than LCMV or MVPCA (Fig. 3.4(a)). However, utilizing all bands yields the best result. This holds also for the SVM (Fig. 3.4(b)). In contrast to KNN, SCBS outperforms AP and is very close to the results when using all bands.

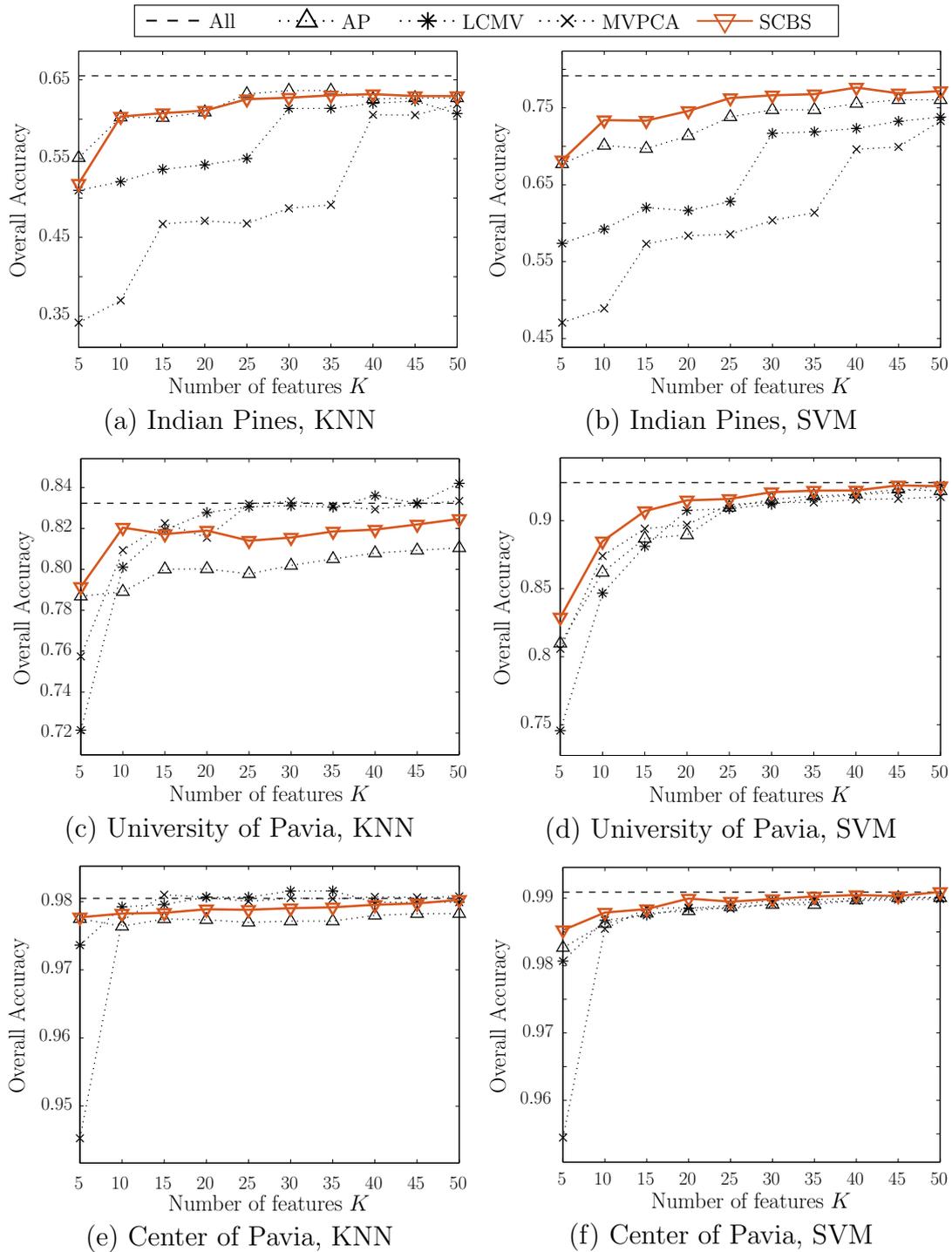


Figure 3.4. Classification results for the Indian Pines, (a) KNN and (b) SVM; University of Pavia, (c) KNN and (d) SVM; and Center of Pavia, (e) KNN and (f) SVM. Using KNN, results comparable with state-of-the-art are achieved. Higher accuracies are achieved by means of SVMs, where the proposed approach outperforms the other methods. The dashed lines show the accuracies using all bands and is therefore constant over the number of features.

Since we have a lot more training samples for the UP data set, we generally obtain higher classification accuracies in this data set. For KNN, LCMV and MVPCA show the best results as depicted in Fig. 3.4(c). Still, SCBS lies close to the other methods and gives more accurate predictions than AP. Using SVMs, SCBS yields the best performance in all cases (Fig. 3.4(d)).

In the CP image, we generally obtain high classification accuracies. With only 5 features, an accuracy of more than 97% is achieved. The results are in line with those of UP, showing that SCBS combined with SVMs yield the highest classification accuracies of the investigated methods.

As IP contains remarkably more bands than the Pavia images, the results suggest that the clustering-based methods, AP and SCBS, are able to deal more efficiently with high-dimensional input data than the correlation and subspace methods, LCMV and MVPCA. Further, as the IP data set has a lower SNR compared with the other images, the performance of LCMV and MVPCA seems to be significantly affected in case of noisy observations.

3.3.4.2 Automatic Number-of-Bands Selection

For estimating the number of features, the content parameter α_{IC} has to be set. By means of cross-validation, we found $\alpha_{IC} = 99.99\%$ yielding high accuracies (c.f. Tab. 3.1 and Tab. 3.2). Despite the high value of α_{IC} , the number of bands is often significantly reduced. For IP, we obtain 182 bands, for CP 54 and UP 50, i.e., reductions of approx. 50% of the size of the original data is achieved in case of the Pavia images. It should be noted, as stated before, that the IP image has a lower SNR than the other images. Further, the classes in this image are similar as they represent vegetation. Thus, the bands contain valuable information necessary to discriminate the classes, resulting in a lower reduction of 9%.

In Tab. 3.1 and Tab. 3.2, we also provide results for the case that morphological operations (opening and closing) [126], denoted by SCBS+M and All+M, are applied after band reduction. These operations exploit the similarity of neighboring pixels to improve the classification accuracy, and, thus, can be considered as spatial filtering. The high accuracies show that, despite reducing the number of bands, high classification accuracies are obtained.

Table 3.1. Classification results of the data sets using SVMs (accuracies in %) where the number of bands is estimated based on the information content.

	OA	IP CA	K	OA	UP CA	K	OA	CP CA	K
All	79.16	85.85	200	92.80	93.20	103	99.09	97.95	102
SCBS	79.35	85.95	182	92.71	93.13	50	99.08	97.92	54
Difference	+0.19	+0.10	-9%	-0.09	-0.07	-51%	-0.01	-0.03	-47%
All+M	91.31	95.57	200	99.20	99.39	103	99.93	99.85	102
SCBS+M	92.49	96.04	182	99.63	99.68	50	99.92	99.84	54
Difference	+1.18	+0.47	-9%	+0.43	+0.29	-51%	-0.01	-0.01	-47%

Table 3.2. Classification results of the data sets (accuracies in %) using KNN where the number of bands is estimated based on the information content.

	OA	IP CA	K	OA	UP CA	K	OA	CP CA	K
All	65.51	75.13	200	83.22	86.47	103	98.05	95.78	104
SCBS	65.50	75.13	182	83.23	86.35	50	98.05	95.77	54
Difference	-0.01	0.00	-9%	+0.01	-0.12	-51%	0.00	-0.01	-47%
All+M	88.34	93.91	200	98.01	98.88	103	99.87	99.69	104
SCBS+M	88.95	93.85	182	97.70	98.60	50	99.84	99.70	54
Difference	+0.61	-0.06	-9%	-0.31	-0.28	-51%	-0.03	+0.01	-47%

3.4 Classification in the Compressive Domain

In this section, we propose a system for pixel-wise Compressive Classification (CC) in hyperspectral imaging. Using this framework enables an efficient acquisition, reducing the amount of captured data and the number of required sensor elements. The pixels of the hyperspectral image are captured in a low-dimensional representation by making use of CS techniques. While most CS literature assumes that the transform matrix is known, we show how this matrix can be learned by means of training samples. Given

the transform matrix, we further show how the sensing matrix can be optimized to fulfill the required incoherence property (Section 2.3.2).³

3.4.1 Compressed Classification

As explained in Section 2.3.2, CS allows to sample the signal of interest below the Nyquist rate. In the following, the signal \mathbf{z} represents the positive-valued spectrum of a pixel with D elements, i.e., $\mathbf{z} \in \mathbb{R}_+^{D \times 1}$. The signal \mathbf{z} is captured by taking only $K < D$ linear measurements [46, 71, 128] which can be expressed as

$$\mathbf{s} = \mathbf{\Phi}\mathbf{z}, \quad (3.5)$$

where $\mathbf{\Phi} \in \mathbb{R}^{K \times D}$ is the sensing or measurement matrix and $\mathbf{s} \in \mathbb{R}^K$ is the observed measurement vector. Note that \mathbf{z} is not observed directly but can be reconstructed by assuming that \mathbf{z} is sparse in some domain, i.e., $\mathbf{z} = \mathbf{\Psi}\mathbf{x}$ with basis $\mathbf{\Psi} \in \mathbb{R}_+^{D \times D}$ and sparse coefficient vector $\mathbf{x} \in \mathbb{R}_+^{D \times 1}$. The reconstruction of \mathbf{z} is then formulated as an optimization problem as explained in Section 2.3.2, for which computationally expensive algorithms have to be employed.

In HSI, we are mainly interested in the content of the image, i.e., not in the image itself. Therefore, it is desirable to skip the step of reconstructing the image and classify the observations in the measurement domain directly. Thus, we interpret the measurement domain as a feature space with linear feature transform $\mathbf{\Phi}$, meaning that the features of the latent signal are directly observed in terms of \mathbf{s} .

Davenport *et al.* present a similar idea in [122]. They propose to transform the training data into the measurement domain and use KNN for the classification, which the authors refer to as the Smashed Filter (SF) due to its similarity with the matched filter. We follow the approach in [129] where the use of SVMs in the measurement domain is demonstrated. For this purpose, a discriminant function, $f_{\text{SVM}} : \mathbf{s} \mapsto y$, with

$$y = \mathbf{w}_{\text{SVM}}^T \mathbf{s} + b_{\text{SVM}}, \quad (3.6)$$

is trained on samples transformed into the compressive domain. During training, the parameters $\mathbf{w}_{\text{SVM}} \in \mathbb{R}^{K \times 1}$ and $b_{\text{SVM}} \in \mathbb{R}$ are estimated by maximizing the margin between the decision boundary and the closest training samples, as explained in Section 2.4.3. Note that the extensions for the SVM introduced in Section 2.4.3 can also be applied here.

³Parts of the work in this section were presented at ICASSP 2014 [101]

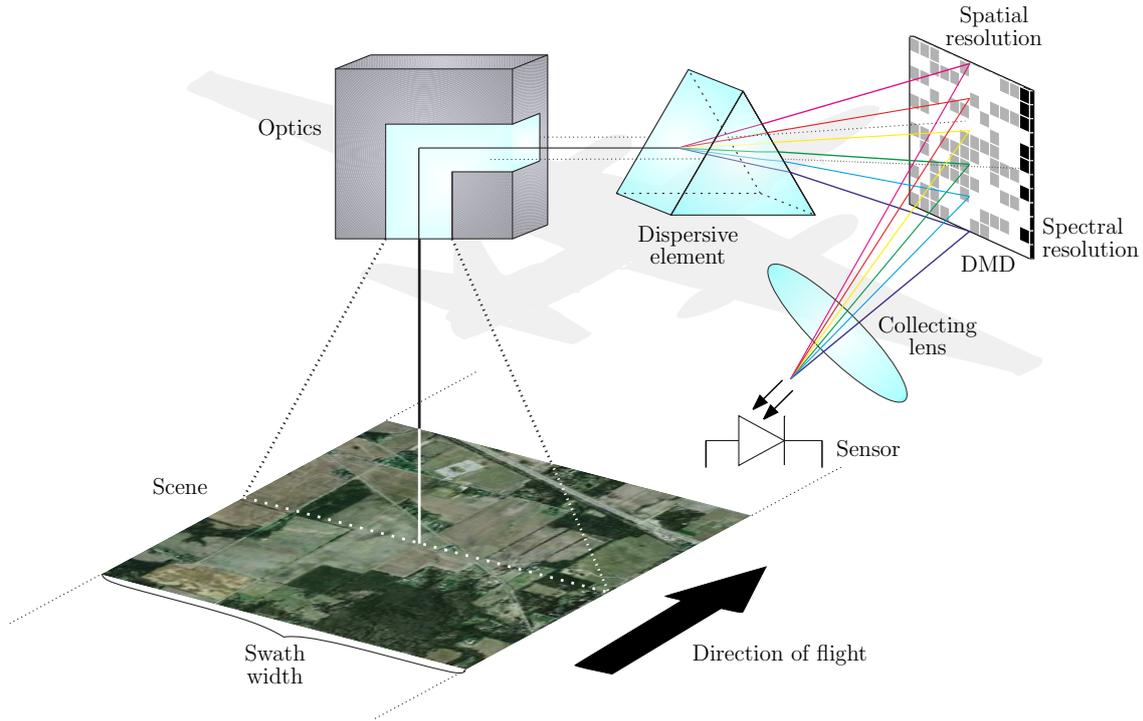


Figure 3.5. Design of a hyperspectral imaging system for the use in Compressed Classification. The incident light is split by a dispersive element. By activating the lines of the DMD separately, different random measurements are taken. In this illustration, the first line of the DMD is activated.

3.4.2 System Design

Let $\mathbf{Z}_{\text{HSI}} \in \mathbb{R}^{N_1 \times N_2 \times D}$ denote the hyperspectral image of size $N_1 \times N_2 \times D$ where N_1 is the number of pixels in cross-track and N_2 in track direction. The number of bands is denoted by D . Pixel-wise CC requires an imaging system that preserves the spatial structure of the scene as otherwise information of different pixels would be mixed. Thus, the low-dimensional acquisition is performed in the spectral domain by taking K measurements of the spectrum $\mathbf{z}_n \in \mathbb{R}^{D \times 1}$ with $1 \leq n \leq N_1 N_2$.

An imaging system based on CC can be designed as illustrated in Fig. 3.5. The scene of interest is measured in scanlines similar to the push-broom technique known from HSI in remote sensing [130]. For remote sensing, this system could be mounted on an airborne device.

As shown in Fig. 3.5, the incident light of a scanline is decomposed into its spectral components by means of a dispersive element. Similar as in [131], the spectral components are reflected from a DMD with $D \times N_1$ elements and detected by a single sensor. Note that we cannot compressively capture all pixels of the scanline at once as the

spatial information would be lost. A possibility to preserve the cross-track resolution is to spread the spatial information into the time domain. For this purpose, only one line of the DMD is activated at a time, i.e., the other lines are set to zero. Hence, a measurement can be considered as the inner product of the pattern represented by a DMD line and the spectral information of the pixel. Having received a measurement, the next line of the DMD is activated to capture the next pixel. For each scanline, this procedure is continued N_1 times. Further, K measurements of each pixel are obtained by choosing K different DMD patterns for each measurement.

3.4.3 Optimization of the Acquisition Process

For classification, and, hence, also for CC, training samples of the classes occurring in the scene of interest are required. The training samples can also be used to optimize the basis matrix of the data, Ψ , and the measurement matrix, Φ . As, according to CS theory, the measurement matrix has to be designed such that it is incoherent to the basis matrix [46, 71, 128], we explain how both matrices can be adapted.

Ideally, both matrices are jointly estimated. Suppose we are given N_T training samples $\mathbf{z}_{T,n} \in \mathbb{R}^{D \times 1}$, with $n = 1, \dots, N_T$, that we stack into an $D \times N_T$ matrix $\mathbf{Z}_T = [\mathbf{z}_{T,1} \ \dots \ \mathbf{z}_{T,N_T}]$ where the corresponding (unknown) sparse coefficients are denoted by $\mathbf{X} \in \mathbb{R}_+^{D \times N_T}$. In a probabilistic framework, the joint estimation of the basis and sensing matrices is formulated as

$$\{\hat{\Psi}, \hat{\Phi}\} = \arg \max_{\Psi, \Phi} \int_{\mathcal{X}} p(\Psi, \Phi, \mathbf{X} | \mathbf{Z}_T) d\mathbf{X},$$

where we marginalize over \mathbf{X} as we are not interested in inferring the coefficients of the training data. While finding an expression for the joint posterior $p(\Psi, \Phi, \mathbf{X} | \mathbf{Z}_T)$ is already challenging, the marginalization is even likely to be intractable. For this reason, we propose to decompose the adaptation process into two steps. In the first step, the basis matrix is estimated from the training data, independent of the sensing matrix. In the second step, we adapt the sensing matrix to the estimated basis such that the matrices become (approximately) incoherent.

3.4.3.1 Learning the Basis

For learning the basis matrix, we seek for a sparse representation of the training data \mathbf{Z}_T . Thus, we reduce the number of columns of the basis matrix to the fixed preset

parameter K_S , with $K_S \leq D$. To estimate the (reduced) latent basis $\hat{\Psi} \in \mathbb{R}_+^{D \times K_S}$ from the training data, we propose a probabilistic modeling of the quantities such that a *maximum-a-posteriori* (MAP) estimate can be obtained. Further, we normalize the rows of the basis matrix, yielding $\tilde{\Psi}$, with $\tilde{\psi}_k := \frac{\psi_k}{\|\psi_k\|_2}$ for $k = 1, \dots, K_S$ [12], for reasons that will be explained below. The reduced sparse coefficients matrix is denoted by $\tilde{\mathbf{X}} \in \mathbb{R}_+^{K_S \times N_T}$.

The normalized, reduced basis is estimated by maximizing the joint posterior of the basis and the coefficients of the training samples,

$$\hat{\Psi} = \arg \max_{\tilde{\Psi}, \tilde{\mathbf{X}}} p(\tilde{\Psi}, \tilde{\mathbf{X}} | \mathbf{Z}_T) \quad (3.7)$$

$$= \arg \max_{\tilde{\Psi}, \tilde{\mathbf{X}}} p(\mathbf{Z}_T | \tilde{\Psi}, \tilde{\mathbf{X}}) p(\tilde{\mathbf{X}}) p(\tilde{\Psi}), \quad (3.8)$$

where our prior belief is that $\tilde{\Psi}$ and $\tilde{\mathbf{X}}$ are independent. In the following, we make two further assumptions. First, we assume that the observations are conditionally independent such that the rows of \mathbf{Z}_T and $\tilde{\mathbf{X}}$ can be modeled separately. Second, we assume that each pixel, \mathbf{z}_n , suffers from additive Gaussian noise, resulting in the following likelihood,

$$p(\mathbf{Z}_T | \tilde{\Psi}, \tilde{\mathbf{X}}) = \prod_{n=1}^{N_T} \mathcal{N}_{\mathbf{z}_{T,n}} \left(\tilde{\Psi} \tilde{\mathbf{x}}_n, \sigma_z^2 \mathbf{I}_D \right), \quad (3.9)$$

with mean $\tilde{\Psi} \tilde{\mathbf{x}}_n$ and noise variance σ_z^2 , where \mathbf{I}_D is the identity matrix of size $D \times D$.

According to CS theory, $\tilde{\mathbf{X}}$ should be sparse. For this reason, we choose a truncated Laplace distribution with zero-mean as sparsity-inducing prior for $\tilde{\mathbf{x}}_n$, $n = 1, \dots, N_T$, such that

$$p(\tilde{\mathbf{X}}) = \prod_{n=1}^{N_T} p(\tilde{\mathbf{x}}_n) \propto \prod_{n=1}^{N_T} \text{Laplace}_{\tilde{\mathbf{x}}_n}(0, b_{\text{Sparse}}) H(\tilde{\mathbf{x}}_n), \quad (3.10)$$

with hyperparameter b_{Sparse} . The operator $H(\mathbf{s})$ returns one if all elements of \mathbf{s} are positive-valued and zero otherwise. For $\tilde{\Psi}$, we assume a flat prior.

Taking the negative logarithm of the posterior in Eq. (3.8) with the likelihood and prior defined in Eq. (3.9) and Eq. (3.10) yields the objective function

$$E_{\text{Sparse}}(\tilde{\Psi}, \tilde{\mathbf{X}}) = \underbrace{\sum_{n=1}^{N_T} \|\mathbf{z}_{T,n} - \tilde{\Psi} \tilde{\mathbf{x}}_n\|_2^2}_{\text{Observation likelihood}} + b_{\text{Sparse}} \underbrace{\sum_{n=1}^{N_T} \|\tilde{\mathbf{x}}_n\|_1}_{\text{Sparsity term}}, \quad (3.11)$$

which needs to be minimized with respect to $\tilde{\Psi}$ and $\tilde{\mathbf{X}}$. When minimizing Eq. (3.11), the following problem can be observed: if $\tilde{\Psi}$ is scaled by a factor $q \in \{r \in \mathbb{R}_+ : r > 1\}$ and $\tilde{\mathbf{X}}$ is scaled by the inverse, $\frac{1}{q}$, the likelihood remains unchanged, but the sparsity term may decrease to zero. Consequently, $\tilde{\mathbf{X}}$ is likely not to become sparse as the influence of the sparsity penalty is reduced [12]. For this reason, as explained above, the columns of $\tilde{\Psi}$ are normalized.

As $\tilde{\Psi}$ is supposed to be a basis matrix, we require (approximately) orthogonal columns, i.e., $\tilde{\Psi}^T \tilde{\Psi} \approx \mathbf{I}_{K_S}$. To enforce orthogonality, we construct a quadratic penalty term,

$$E_{\text{Orth}}(\tilde{\Psi}) = \|\tilde{\Psi}^T \tilde{\Psi} - \mathbf{I}_{K_S}\|_{\text{F}}^2. \quad (3.12)$$

Combining the objective function in Eq. (3.11) and the orthogonality penalty in Eq. (3.12) results in

$$E(\tilde{\Psi}, \tilde{\mathbf{X}}) = \sum_{n=1}^{N_T} \|\mathbf{z}_{T,n} - \tilde{\Psi} \tilde{\mathbf{x}}_n\|_2^2 + b_{\text{Sparse}} \sum_{n=1}^{N_T} \|\tilde{\mathbf{x}}_n\|_1 + b_{\text{Orth}} \|\tilde{\Psi}^T \tilde{\Psi} - \mathbf{I}_{K_S}\|_{\text{F}}^2, \quad (3.13)$$

where b_{Orth} is a regularization parameter that controls the orthogonality constraint.

Estimates of the sparse coefficient matrix, $\hat{\mathbf{X}}$, and the basis matrix, $\hat{\Psi}$, are obtained by minimizing Eq. (3.13) with respect to $\tilde{\Psi}$ and $\tilde{\mathbf{X}}$. For this purpose, a gradient descent method is used where $\tilde{\Psi}$ and $\tilde{\mathbf{X}}$ are iteratively updated in turns. Since we constrain the basis and the coefficients to be positive valued, we use a gradient descent algorithm with a multiplicative update rule (c.f. [12, 132]),

$$\tilde{\mathbf{X}}^{(l+1)} = \tilde{\mathbf{X}}^{(l)} \odot \frac{\left[\nabla_{\tilde{\mathbf{X}}} E(\tilde{\mathbf{X}} = \tilde{\mathbf{X}}^{(l)}) \right]^-}{\left[\nabla_{\tilde{\mathbf{X}}} E(\tilde{\mathbf{X}} = \tilde{\mathbf{X}}^{(l)}) \right]^+} \quad (3.14)$$

$$\tilde{\Psi}^{(l+1)} = \tilde{\Psi}^{(l)} \odot \frac{\left[\nabla_{\tilde{\Psi}} E(\tilde{\Psi} = \tilde{\Psi}^{(l)}) \right]^-}{\left[\nabla_{\tilde{\Psi}} E(\tilde{\Psi} = \tilde{\Psi}^{(l)}) \right]^+} \quad (3.15)$$

where l denotes the l th iteration of the algorithm, \odot denotes the Hadamard product and $[\cdot]^+$ and $[\cdot]^-$ select the positive and negative terms of the partial derivatives. The gradients are given as

$$\nabla_{\tilde{\mathbf{X}}} E(\tilde{\mathbf{X}} = \tilde{\mathbf{X}}^{(l)}) = \left. \frac{\partial E(\tilde{\Psi}, \tilde{\mathbf{X}})}{\partial \tilde{\mathbf{X}}} \right|_{\tilde{\mathbf{x}}=\tilde{\mathbf{x}}^{(l)}} \quad (3.16)$$

$$\nabla_{\tilde{\Psi}} E(\tilde{\Psi} = \tilde{\Psi}^{(l)}) = \left. \frac{\partial E(\tilde{\Psi}, \tilde{\mathbf{X}})}{\partial \tilde{\Psi}} \right|_{\tilde{\psi}=\tilde{\psi}^{(l)}}. \quad (3.17)$$

In contrast to conventional gradient ascent, this update rule has the advantage of selecting the step width automatically. The derivatives in Eq. (3.16) and Eq. (3.17) can be derived as follows:

$$\begin{aligned} \frac{\partial E(\tilde{\Psi}, \tilde{\mathbf{X}})}{\partial \tilde{\mathbf{X}}} &= -2 \sum_{n=1}^{N_T} \tilde{\Psi}^T (\mathbf{z}_{T,n} - \tilde{\Psi} \tilde{\mathbf{x}}_n) + N_T b_{\text{Sparse}} \\ \frac{\partial E(\tilde{\Psi}, \tilde{\mathbf{X}})}{\partial \tilde{\psi}_k} &= 2 \frac{\tilde{x}_{k,n}}{\|\tilde{\psi}_k\|_2} \left(-\mathbf{z}_n + \sum_{k'=1}^K \tilde{x}_{k',n} \tilde{\psi}_{k'} + \mathbf{z}_{T,n} (\tilde{\psi}_k \tilde{\psi}_k^T)^T - (\tilde{\psi}_k \tilde{\psi}_k^T)^T \sum_{k'=1}^K \tilde{x}_{k',n} \tilde{\psi}_{k'} \right) \\ &\quad + \frac{4b_{\text{Orth}}}{\|\tilde{\psi}_k\|_2} \left(\sum_{k'=1}^K \tilde{\psi}_{k'} \right) \left(\sum_{k'=1}^K (\tilde{\psi}_{k'})^T \tilde{\psi}_{k'} - 1 - (\tilde{\psi}_k \tilde{\psi}_k^T)^T \sum_{k'=1}^K (\tilde{\psi}_{k'})^T \tilde{\psi}_{k'} \right. \\ &\quad \left. + (\tilde{\psi}_k \tilde{\psi}_k^T)^T \right) \end{aligned}$$

where we used $\tilde{\psi}_k = \frac{\hat{\psi}_k}{\|\hat{\psi}_k\|_2}$, $k = 1, \dots, K_S$. The derivations of the derivatives can be found in Appendix A.

Note that the choice of parameters b_{Sparse} , b_{Orth} and K_S in Eq. (3.13) has a significant influence on the result. Since we are interested in maximizing the classification accuracy, we optimize these parameters with respect to the classification accuracy by means of cross-validation [95].

3.4.3.2 Learning the Sensing Matrix

Given a basis, $\tilde{\Psi} \in \mathbb{R}^{D \times K_S}$, the sensing matrix $\tilde{\Phi} \in \mathbb{R}^{K \times D}$ has to be adjusted such that the correlation between the columns of $\tilde{\Phi} \tilde{\Psi}$ is minimized. This is similar to finding an orthogonal basis matrix, as formulated in Eq. (3.12). Thus, we need to solve

$$(\tilde{\Phi} \tilde{\Psi})^T \tilde{\Phi} \tilde{\Psi} \approx \mathbf{D}_{K_S}, \quad (3.18)$$

where \mathbf{D}_{K_S} is a $K_S \times K_S$ diagonal matrix. For simplicity, assume that the columns of $\tilde{\Phi} \tilde{\Psi}$ are normalized so that $\mathbf{D}_{K_S} = \mathbf{I}_{K_S}$. Multiplying Eq. (3.18) by $\tilde{\Psi}$ from both sides yields

$$(\tilde{\Phi} \tilde{\Psi})^T \tilde{\Phi} \tilde{\Psi} = \mathbf{I}_{K_S} \quad (3.19)$$

$$(\tilde{\Phi} \tilde{\Psi} \tilde{\Psi}^T)^T \tilde{\Phi} \tilde{\Psi} \tilde{\Psi}^T = \tilde{\Psi} \tilde{\Psi}^T. \quad (3.20)$$

Consider for the moment the case when $K_S = D$ and $K = D$, i.e., the sensing and basis matrices become square. Thus, the optimal solution for $\tilde{\Phi}$ can be found by substituting $\tilde{\Psi} \tilde{\Psi}^T$ by its eigendecomposition. Then, we can write

$$\tilde{\Psi} \tilde{\Psi}^T = \mathbf{V} \Lambda \mathbf{V}^T$$

with $\mathbf{V} \in \mathbb{R}^{D \times D}$ containing the eigenvectors and the diagonal matrix $\mathbf{\Lambda} \in \mathbb{R}^{D \times D}$ the eigenvalues. The optimal solution for the measurement matrix is given by

$$\hat{\Phi}_{\text{Opt}} = \mathbf{\Lambda}^{-1/2} \mathbf{V}^T.$$

Based on this result, we estimate a matrix $\hat{\Phi}'$ of rank K that approximates $\hat{\Phi}_{\text{Opt}}$ by solving

$$\hat{\Phi}' = \arg \min_{\tilde{\Phi}'} \|\tilde{\Phi}' - \hat{\Phi}_{\text{Opt}}\|_{\text{F}}^2 \quad \text{s.t.} \quad \text{rank}(\tilde{\Phi}') = K \quad (3.21)$$

where $\tilde{\Phi}'$ is a $D \times D$ matrix of rank K . Eckart and Young [133] provide a closed form solution for Eq. (3.21), known as the *Eckart-Young-Theorem*. In fact, we use only the top K eigenvalues of $\hat{\Phi}_{\text{Opt}}$ to estimate $\hat{\Phi}'$, i.e.,

$$\hat{\Phi}' = \lceil \mathbf{\Lambda}^{-1/2} \rceil^K \mathbf{V}^T, \quad (3.22)$$

where $\lceil \cdot \rceil^K$ chooses only the top K values and replaces the remaining elements with zeros. Note that, as in CS, we only need to make sure that the number of measurements is greater than the sparsity, $K \geq K_S$, as the optimization problem would be underdetermined, otherwise. Finally, we obtain the learned sensing matrix $\hat{\Phi} \in \mathbb{R}_+^{K \times D}$.

3.4.4 Results

We evaluate the proposed methods using the Indiana's Indian Pines (IP) [124] and the University of Pavia (UP) [125] data sets based on the Overall Accuracy (OA) and Class Accuracy (CA) [100]. In contrast to the experiments in Section 3.3.4 we do not remove the noisy bands as we simulate an acquisition approach where we do not know *a priori* which bands are noisy. Of course, this makes classification even more challenging.

In order to show the performance of the presented methods, two classifiers are used: the KNN [91] and SVMs [127]. Note that using KNN for CC is equivalent to the Smashed Filter [122]. In the following, we show the OA of CC, Adaptive Compressed Classification (ACC), and conventional classification (All), using all available data, for different subsampling rates, i.e., reduced numbers of bands K . Note that, if we use all available data, the result is independent of K and can be considered as a baseline. CC uses a (random) binary sensing matrix. In contrast, ACC optimizes the sensing as well as the measurement matrices as described in Section 3.4.3.

In each setup we compute the correct classification accuracies by randomly splitting the data set into 10% training and 90% test data. The results are averaged over 50

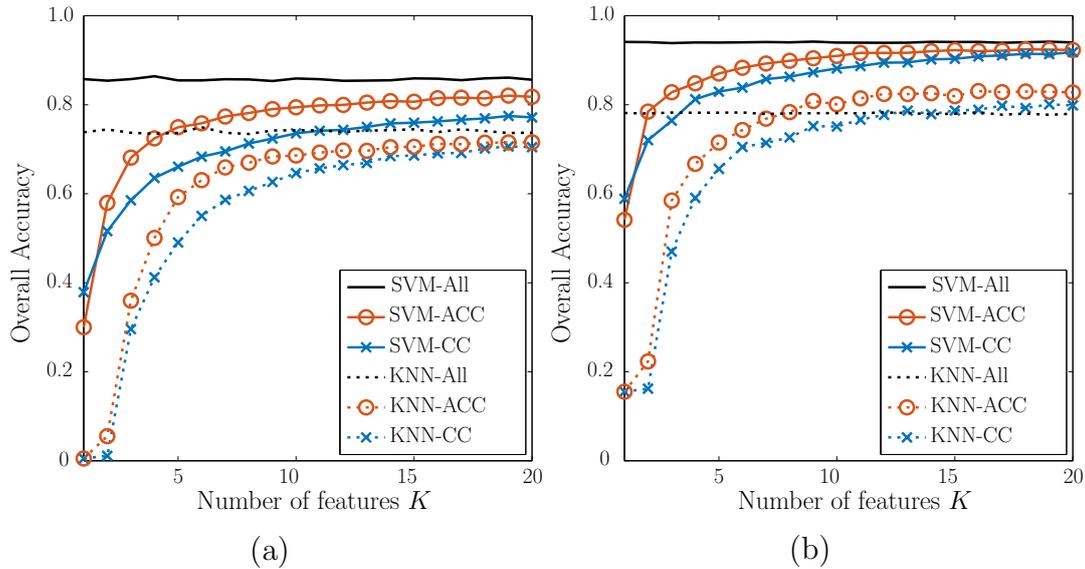


Figure 3.6. Classification results for (a) Indian Pines and (b) University of Pavia. The basis and sensing matrix adaptations yield significant performance improvements in all cases. In combination with SVMs, KNN is easily outperformed even if all bands are considered.

repetitions, using different test and training sets and sensing matrices in each run. Note that the Monte Carlo runs lead to some fluctuations in the results for All. In order to estimate the parameters b_{Sparse} , b_{Orth} and K for ACC, cross-validation is used resulting in the following parameter settings: $b_{\text{Sparse}} = 0.75$, $b_{\text{Orth}} = 0.5$, $K = 100$ (IP) and $b_{\text{Sparse}} = 0.25$, $b_{\text{Orth}} = 0.25$, $K = 50$ (UP).

The results for the data sets are shown in Fig. 3.6. Most important, we observe that even for a low number of measurements, CC as well as ACC yield results close to the conventional classification accuracy. Capturing only 10% of the data gives on average an accuracy of 77.15% for the IP and 88.17% for the UP using SVMs. When KNN is used for the Indian Pines, CC and ACC provide even more accurate results than All. Using the proposed adaptation of the basis and measurement matrices, the results are improved by 3.5% to 7% on average. This effect becomes more apparent if we decrease the number of measurements to, e.g., only three measurements, which leads to an improvement of more than 6.5% (UP) and 9.3% (IP). Tab. 3.3 provides more detailed results for IP and UP where only 10% of the spectral information of the original data is used for classification.

Table 3.3. Classification accuracies (in %) of the data sets where only 10% of original samples have been captured.

	Indian Pines				University of Pavia			
	KNN		SVM		KNN		SVM	
	OA	CA	OA	CA	OA	CA	OA	CA
All	73.73	75.09	85.77	84.89	77.99	73.96	94.03	92.23
ACC	71.57	72.83	81.80	81.81	82.77	79.44	90.98	88.43
CC	70.45	71.13	77.21	77.53	79.94	76.90	88.17	85.11

3.5 Conclusions

In this chapter, we have present two different approaches to learn low-dimensional representations for the classification of hyperspectral images. In the first approach, we have considered the spectral information of each pixel as a feature vector. By means of a clustering approach and dimensionality reduction techniques, we have selected the most relevant features, resulting in a band selection algorithm for hyperspectral imaging. The results based on real images show that the number of features can be reduced by 10 % to 50 %, still yielding high classification accuracies. Further, we have presented a framework that allows to capture the hyperspectral image in a low-dimensional domain directly based on Compressive Sensing. As we are not interested in the reconstruction of the original image, we consider the captured data as representing the original data in a feature space and classify them directly. To improve the classification accuracy, the sensing and basis matrices are learned from the observations. The results of this approach show high classification accuracies, which are further increased by means of learning the measurement and basis matrices.

Chapter 4

Hyperspectral Unmixing via Bayesian Nonparametric Feature Learning

In the previous chapter, we have proposed a feature selection algorithm for hyperspectral images and a low-dimensional acquisition approach for direct classification, where costly reconstructing of the image can be avoided. This chapter is concerned with Hyperspectral Unmixing (HSU), which describes the problem of jointly estimating the endmembers and their abundances present in the scene of interest. The endmembers can be understood as the raw materials present in the scene, while the abundances describe the fractions of which the endmembers occur in each pixel of the captured image.

We give an introduction into HSU and motivate the proposed approach in Section 4.1. An overview of the state of the art in HSU is given in Section 4.2. Our contributions are highlighted in Section 4.3. Based on [134] and [135], we provide a Bayesian nonparametric model for HSU in Section 4.4 and show in Section 4.5 how to perform inference in this model. Section 4.6 provides results on synthetic as well as real data. In Section 4.7, we comment on the results and provide insights for future directions.¹

4.1 Motivation and Introduction

Especially in remote sensing, hyperspectral images can often be only captured at a low spatial resolution. Thus, the spectrum in a pixel is likely to represent a mixture of different materials. However, most algorithms used for data analysis, such as in classification, assume pure pixels, i.e., each pixel is assumed to represent a single material. For this reason, HSU is an important task for the analysis of hyperspectral images, revealing the endmembers and their abundances present in the scene of interest.

Although there already exist many methods and algorithms that aim at solving the unmixing problem, a key limitation of most approaches is their assumption that the number of endmembers present in the scene is known *a priori*. However, this is rarely

¹This chapter has served as basis for the journal article: J. Hahn and A. M. Zoubir, “Bayesian Nonparametric Unmixing of Hyperspectral Images”, submitted to *IEEE Transactions on Geoscience and Remote Sensing*, 2016.

the case in practice and, thus, this number has to be estimated by an expert. If the estimated number is incorrect, most methods will fit the observed data into an incorrect model, which is likely to yield poor results. Especially an underestimate of the number is critical, as then endmembers present in the scene are simply not extracted and remain undiscovered.

The task of unmixing can be also considered as a feature learning problem where the features underlie certain constraints. Recalling that the BNFL framework, which is based on the IBP, enables to infer the number of latent features, we use this framework to jointly infer the number of endmembers, the endmembers, and the abundances. As opposed to most existing work, this framework also allows to infer the number of endmembers present in the image.

4.2 State of the Art

Today, many algorithms exist for HSU. An excellent overview is given in [7]. Most algorithms solve the unmixing problem by, first, extracting the endmembers and, second, estimating the abundances, e.g., Pixel Purity Index (PPI) [136], N-FIND-R [137], and Vertex Component Analysis (VCA) [138]. In contrast, Bayesian methods are often based on generative models, enabling them to perform both steps jointly. In [134], a Bayesian framework is presented for jointly inferring the endmembers and abundances. For this purpose, the authors investigate different priors for the endmembers, which are motivated by regularization terms of existing nonprobabilistic unmixing methods. A different Bayesian framework is proposed in [135], where the authors exploit that the abundances lie in a subspace. Other examples of algorithms that are able to jointly estimate the abundances and endmembers are Iterated Constrained Endmembers (ICE) [139], Minimum Volume Transform (MVT) [140], Minimum-Volume Enclosing Simplex (MVES) [141] and NMF [132, 142]. These methods have in common that they assume a linear relation between the endmembers and abundances, motivated by a macroscopic illumination model [7, 143]. Recent work investigates also nonlinear models [144–147], required for a microscopic model accounting for nonlinear illumination effects [7, 148]. Further, semi-supervised approaches have been explored, where the endmembers in the scene are selected from a dictionary instead of being learned [149, 150].

There is only few work that is concerned with estimating the number of endmembers, where most approaches are based on subspace methods [7]. Classic methods

for model-order selection, such as Akaike Information Criterion (AIC) [151] and Minimum Description Length (MDL) [152] do not work well in the context of HSU due to their assumptions on the noise [116]. Thus, several approaches designed specifically for HSI have been proposed in the recent past. The probably most prominent method is VD [116], where information theoretic criteria are utilized together with a Neyman-Pearson test to detect the number of endmembers. Hyperspectral Signal Subspace Identification by Minimum Error (HySime) [153] estimates the signal subspace by minimizing the projection errors of the signal and noise subspace. The dimensionality of the signal subspace is then considered as an estimate for the number of endmembers present in the scene. A similar approach is Eigenvalue Likelihood Maximization (ELM) [154], which compares the correlation and covariance matrix of the spectra to infer the number of endmembers. A geometric approach based on nearest neighbors is proposed in [155], where the dimensionality of a manifold is estimated to learn the number of endmembers. Due to the sensitivity to noise of the method, a denoised version of the algorithm, Denoised Hyperspectral Intrinsic Dimensionality Estimation With Nearest-Neighbor Distance Ratios (HideNN), is also presented. In [156], Sparsity-Promoting ICE (SPICE) is presented which aims at estimating the endmembers, their fractional abundances and the number of endmembers jointly. SPICE extends the ICE algorithm by placing a sparsity promoting prior on the abundances. After applying a thresholding scheme to the abundances, endmembers are pruned if not present in the scene.

4.3 Contribution

We consider HSU as a constrained feature learning problem, where the features represent the endmembers and the coefficients of the features the abundances. A BFL-based framework for HSU has been proposed in [134], where the number of features needs to be set by the user. As explained, inferring the number of latent endmembers is an important problem in HSU. Therefore, we follow the approach in [75] and extend the Bayesian framework in [134] by placing an IBP prior on the activations of the endmembers, resulting in a Bayesian nonparametric model [77]. As explained in Section 2.3.3.1, the IBP provides means for inferring the number of present endmembers in the scene. Thus, our proposed algorithm, Bayesian Nonparametric Unmixing (BNU), allows for the joint inference of the endmembers, their abundances, and also the number of endmembers, in contrast to most existing HSU algorithms.

4.4 Bayesian Nonparametric Unmixing Model

We consider a linear unmixing model for the observed spectra, $\mathbf{z}_n \in \mathbb{R}^{D \times 1}$, $n = 1, \dots, N_z$, of a hyperspectral image with N_z pixels, D bands, and additive noise, $\mathbf{e}_n \in \mathbb{R}^{D \times 1}$,

$$\mathbf{z}_n = \mathbf{F}\mathbf{s}_n + \mathbf{e}_n, \quad (4.1)$$

where $\mathbf{F} \in \mathbb{R}_+^{D \times K}$ are the endmembers and $\mathbf{s}_n \in [0, 1]^{K \times 1}$ the corresponding abundances. The number of endmembers is denoted by K . Comparing Eq. (4.1) with the BFL model in Section 2.3.3 reveals the similarity between linear unmixing and the linear latent feature model. Thus, the endmembers can be considered as positive-valued features and the abundances as constrained feature coefficients. In particular, the abundances are required to fulfill the *additivity constraint*, i.e., $\sum_{k=1}^K s_{n,k} = 1$, and the *positivity constraint*, i.e., $s_{n,k} \geq 0$, with $n = 1, \dots, N_z$ and $k = 1, \dots, K$, as they represent the fractions of which the endmembers occur in each pixel. As in BFL, the noise, \mathbf{e}_n , is assumed to be completely i.i.d. Gaussian distributed, i.e., $p(\mathbf{e} | \sigma_z) = \mathcal{N}_{\mathbf{e}}(0, \sigma_z^2 \mathbf{I})$ with variance σ_z^2 . Although this model does not capture correlated noise, it has been widely used in unmixing models, e.g., in [135, 157].

As explained, HSU can be considered as a feature learning problem. Hence, we can model the unmixing task by means of the BNFL framework, yielding the BNU algorithm. Thus, we assume that the endmember matrix, \mathbf{F} , is composed of a binary activation matrix $\mathbf{A} \in \{0, 1\}^{D \times K}$ and a weighting matrix $\mathbf{W} \in \mathbb{R}_+^{D \times K}$ as suggested in [75] and detailed in Section 2.3.3, i.e.,

$$\mathbf{F} = \mathbf{A} \odot \mathbf{W}, \quad (4.2)$$

where \odot represents the element-wise matrix multiplication. Following a nonparametric approach, we model \mathbf{A} as an IBP. We want to highlight that the samples drawn from an IBP can be dense, though the IBP models a sparse matrix. This is explained in Section 2.3.3.1.

In the following, we detail the components of the proposed hierarchical Bayesian nonparametric model for spectral unmixing.

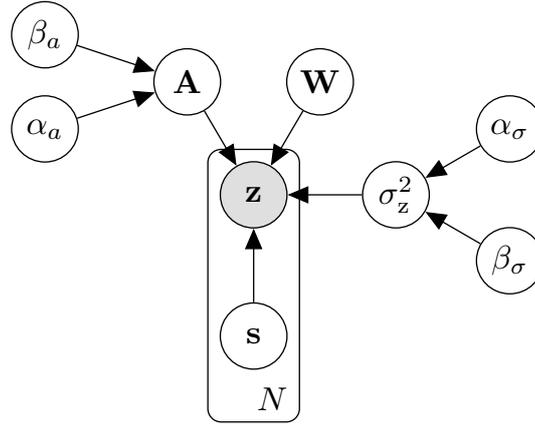


Figure 4.1. Graphical model of the hierarchical Bayesian Nonparametric Unmixing (BNU) model. Only the spectra $\mathbf{z}_n, n = 1, \dots, N_z$, are observed, the other variables are latent and need to be inferred.

4.4.1 Observation Likelihood

We assume that the observed data is conditionally independently distributed and corrupted by additive Gaussian noise. Hence, the likelihood is given as

$$p(\mathbf{Z} | \mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z^2) = \prod_{n=1}^{N_z} \mathcal{N}_{z_n}((\mathbf{A} \odot \mathbf{W}) \mathbf{s}_n, \sigma_z^2 \mathbf{I}), \quad (4.3)$$

with $\mathbf{Z} = [\mathbf{z}_1 \ \dots \ \mathbf{z}_{N_z}]$ and $\mathbf{S} = [\mathbf{s}_1 \ \dots \ \mathbf{s}_{N_z}]$.

In practice, the pixels of the image may suffer also from different lighting conditions that may lead to scattering effects which are not captured by a Gaussian noise model. Deriving a suitable model is challenging, and inference is likely to be intractable. To analyze the effect of varying light conditions, we simulate multiplicative noise on the abundances in Section 4.6.

4.4.2 Prior for the Noise Variance

The variable σ_z^2 denotes the variance of the Gaussian noise. Hence, a suitable prior for σ_z^2 is the Inverse-Gamma distribution with parameters α_σ and β_σ ,

$$p(\sigma_z^2 | \alpha_\sigma, \beta_\sigma) = \text{IGa}_{\sigma_z^2}(\alpha_\sigma, \beta_\sigma).$$

Further, we assume that α_σ and β_σ follow Gamma distributions with $p(\alpha_\sigma) = \text{Ga}_{\alpha_\sigma}(h_{\alpha_\sigma}^{(1)}, h_{\alpha_\sigma}^{(2)})$ and $p(\beta_\sigma) = \text{Ga}_{\beta_\sigma}(h_{\beta_\sigma}^{(1)}, h_{\beta_\sigma}^{(2)})$, respectively, where $h_{\alpha_\sigma}^{(1)}, h_{\alpha_\sigma}^{(2)}, h_{\beta_\sigma}^{(1)}$ and $h_{\beta_\sigma}^{(2)}$ denote the respective hyperparameters.

4.4.3 Prior for the Abundances

Recalling that the prior on \mathbf{S} needs to fulfill the additivity and positivity constraints, we use independent Dirichlet distributions to model the columns of \mathbf{S} . Thus, the abundances of the pixels are assumed to be i.i.d. yielding

$$p(\mathbf{S}) = \prod_{n=1}^{N_z} p(\mathbf{s}_n) = \prod_{n=1}^{N_z} \text{Dir}_{s_{n,1}, \dots, s_{n,K}}(\alpha_{s,1}, \dots, \alpha_{s,K}).$$

We set the hyperparameters $\alpha_{s,k} = 1$ for $k = 1, \dots, K$, making the prior uniform under the additivity and positivity constraints. The uniform distribution has the advantage that no preferences on the different endmembers are imposed. Moreover, this assumption allows for efficient sampling as explained in Section 4.5.

4.4.4 Prior for the Endmember Weights and Activations

Following [134], we choose the distance prior for the endmember weights, \mathbf{W} , with hyperparameter γ_w . This prior can be interpreted as a probabilistic version of the volume regularization proposed in [139] which is based on the Euclidean distance. Further, we use the Heaviside step function to express the positivity constraint on the endmembers, yielding the prior of \mathbf{W} [134],

$$p(\mathbf{W}) \propto \exp\left\{-\gamma_w \sum_{k=1}^K \|\mathbf{w}_k - \frac{1}{K} \sum_{k'=1}^K \mathbf{w}_{k'}\|_2^2\right\} H(\mathbf{W}), \quad (4.4)$$

where \mathbf{w}_k is the k th row of \mathbf{W} .

A drawback of this prior is that the hyperparameter γ_w cannot be inferred from the observations. This is due to the fact that the normalization of the prior $p(\mathbf{W})$ is unknown, which is required to derive the conditional $p(\gamma_w)$ needed for sampling. Thus, γ_w needs to be set *a priori*. We choose this prior as it has shown good performance in the experiments, despite this drawback.

As explained, the feature activation matrix, \mathbf{A} , is modeled as IBP as described in Section 2.3.3.1.

4.4.5 Joint Posterior Distribution

The posterior given the observed spectra, \mathbf{Z} , can be expressed as

$$\begin{aligned} p(\mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z^2, \alpha_\sigma, \beta_\sigma, \alpha_a, \beta_a | \mathbf{Z}) &\propto p(\mathbf{Z} | \mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z^2) p(\sigma_z^2 | \alpha_\sigma, \beta_\sigma) \\ &\quad \times p(\mathbf{S}) p(\mathbf{W}) P(\mathbf{A} | \alpha_a, \beta_a) \\ &\quad \times p(\alpha_\sigma) p(\beta_\sigma) p(\alpha_a) p(\beta_a). \end{aligned} \quad (4.5)$$

The structure of the conditionals, i.e., the conditional independences between the variables, is depicted as a graphical model in Fig. 4.1.

4.5 Inference

As the joint posterior in Eq. (4.5) cannot be derived analytically, we use samples to approximate the posterior, which are generated by means of Gibbs sampling [61]. For the Gibbs Sampler, the conditionals of the variables are derived in the following. We use the bar symbol ($-$) to represent the set of conditional variables, which contains all variables except the one that is sampled.

4.5.1 Sampling the Noise Variance

As the hyperpriors for α_σ and β_σ are conjugate to the prior, the conditional $p(\sigma_z^2 | -)$ is Inverse-Gamma distributed,

$$\begin{aligned} p(\sigma_z^2 | -) &\propto p(\mathbf{Z} | \mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z) p(\sigma_z^2 | \alpha_\sigma, \beta_\sigma) \\ &\propto \text{IGa}_{\sigma_z^2} \left(\alpha_\sigma + \frac{N_z D}{2}, \beta_\sigma + \frac{1}{2} \sum_{n=1}^{N_z} \sum_{d=1}^D \left(z_{n,d} - \sum_{k=1}^K a_{d,k} w_{d,k} s_{k,n} \right)^2 \right), \end{aligned}$$

where sampling from an Inverse-Gamma distribution is easily performed. The derivation can be found in Appendix B.2. For the hyperparameters α_σ and β_σ , the conditionals are

$$\begin{aligned} p(\alpha_\sigma | -) &\propto p(\sigma_z^2 | \alpha_\sigma, \beta_\sigma) p(\alpha_\sigma | h_{\alpha_\sigma}^{(1)}, h_{\alpha_\sigma}^{(2)}) \\ &\propto \text{IGa}_{\sigma_z^2}(\alpha_\sigma, \beta_\sigma) \text{Ga}_{\alpha_\sigma}(h_{\alpha_\sigma}^{(1)}, h_{\alpha_\sigma}^{(2)}), \end{aligned}$$

and, analogously,

$$\begin{aligned} p(\beta_\sigma | -) &\propto p(\sigma_z^2 | \alpha_\sigma, \beta_\sigma) p(\beta_\sigma | h_{\beta_\sigma}^{(1)}, h_{\beta_\sigma}^{(2)}) \\ &\propto \text{IGa}_{\sigma_z^2}(\alpha_\sigma, \beta_\sigma) \text{Ga}_{\beta_\sigma}(h_{\beta_\sigma}^{(1)}, h_{\beta_\sigma}^{(2)}). \end{aligned}$$

For sampling α_σ and β_σ , we use an independent Metropolis-Hastings algorithm with a Gamma proposal distribution. The parameters of the Gamma distributions are chosen as such that the mean of the distribution is given by the previous value and the variance is set to a predefined parameter.

4.5.2 Sampling the Abundances

Under the positivity and additivity constraints, the prior for the abundances is given as a uniform distribution. Thus, the conditional, $p(\mathbf{S} | -)$, is proportional to the likelihood in Eq. (4.3). Due to independence of the pixels, the conditional for the abundance of the n th pixel, \mathbf{s}_n , can be written as a Gaussian distribution,

$$\begin{aligned} p(\mathbf{s}_n | -) &\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{d=1}^D (z_{n,d} - \mathbf{f}_d \mathbf{s}_n^T)^2\right\} \\ &\propto \mathcal{N}_{\mathbf{s}_n}(\boldsymbol{\mu}_{\mathbf{s}_n}, \boldsymbol{\Sigma}_{\mathbf{s}_n}), \end{aligned}$$

with mean $\boldsymbol{\mu}_{\mathbf{s}_n}$ and covariance matrix $\boldsymbol{\Sigma}_{\mathbf{s}_n}$ for $n = 1, \dots, N_z$,

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{s}_n} &= \left(\sum_{d=1}^D \mathbf{f}_d^T \mathbf{f}_d \right)^{-1} \sum_{d=1}^D \mathbf{f}_d z_{n,d}^T, \\ \boldsymbol{\Sigma}_{\mathbf{s}_n} &= \sigma_z^2 \mathbf{I}_K \left(\sum_{d=1}^D \mathbf{f}_d^T \mathbf{f}_d \right)^{-1}, \end{aligned}$$

with \mathbf{I}_K denoting the identity matrix of size K . Thus, we need to sample from a multivariate Gaussian with the constraints that $\sum_{k=1}^K s_{k,n} = 1$ and $0 \leq s_{k,n}$ for all $k = 1, \dots, K$. Sampling from a constrained multivariate Gaussian can be accomplished by Gibbs Sampling [135]. Note that the hyperparameters of $p(\mathbf{S})$, $\alpha_{s,k}$ with $k = 1, \dots, K$, are required to be one. Otherwise, the prior is no longer a (constrained) uniform distribution, which would make sampling the conditional $p(\mathbf{s}_n | -)$ computationally expensive as we would need to resort to less efficient sampling strategies.

4.5.3 Sampling the Endmember Weights

The conditional of \mathbf{W} , $p(\mathbf{W} | -)$, is proportional to the likelihood and the prior, i.e.,

$$\begin{aligned} p(\mathbf{W} | -) &\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \left\| \mathbf{z}_n - \sum_{k'=1}^K s_{k',n} (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) \right\|_2^2 \right. \\ &\quad \left. - \gamma_w \sum_{k'=1}^K \left\| \mathbf{w}_{k'} - \frac{1}{K} \sum_{k''=1}^K \mathbf{w}_{k''} \right\|_2^2 \right\} H(\mathbf{W}). \end{aligned} \tag{4.6}$$

In [134], it is shown that the conditional of the k th feature weight vector, \mathbf{w}_k , is given as

$$p(\mathbf{w}_k | -) \propto \mathcal{TN}_{\mathbf{w}_k}(\boldsymbol{\mu}_{\mathbf{w}_k}, \boldsymbol{\Sigma}_{\mathbf{w}_k}).$$

with $\mathcal{TN}_{\mathbf{w}_k}(\boldsymbol{\mu}_{\mathbf{w}_k}, \boldsymbol{\Sigma}_{\mathbf{w}_k})$ denoting the truncated Gaussian distribution of variable \mathbf{w}_k with mean $\boldsymbol{\mu}_{\mathbf{w}_k}$ and covariance matrix $\boldsymbol{\Sigma}_{\mathbf{w}_k}$. Recalling that the feature activations and weights are linked by an element-wise multiplication (Eq. (4.2)), the conditional covariance matrix, $\boldsymbol{\Sigma}_{\mathbf{w}_k}$, and the mean vector, $\boldsymbol{\mu}_{\mathbf{w}_k}$, can be expressed as

$$\begin{aligned} \boldsymbol{\Sigma}_{\mathbf{w}_k}^{-1} &= \frac{1}{\sigma_z^2} \sum_{n=1}^N s_{k,n}^2 \text{diag}(\mathbf{a}_k) + 2\gamma_w \left(1 - \frac{1}{K}\right) \mathbf{I}_D, \\ \boldsymbol{\mu}_{\mathbf{w}_k} &= \boldsymbol{\Sigma}_{\mathbf{w}_k}^{-1} \left(\frac{1}{\sigma_z^2} \sum_{n=1}^N s_{k,n} (\mathbf{z}_n - \sum_{\substack{k'=1 \\ k' \neq k}}^K s_{k',n} (\mathbf{a}_{k'} \odot \mathbf{w}_{k'})) \odot \mathbf{a}_k - \gamma_w \frac{2}{K} \sum_{\substack{k'=1 \\ k' \neq k}}^K \mathbf{w}_{k'} \right). \end{aligned}$$

We use the method described in [158] to draw samples from a truncated Gaussian.

For completeness, the derivation of the conditional is given in Appendix B.1.1.

4.5.4 Sampling the Endmember Activations

Sampling the activations consists of two steps. First, the active columns are updated, i.e., the d th band of the k th endmember is set active with probability

$$P(a_{k,d} = 1 | -) \propto p(\mathbf{z}_d | \mathbf{f}_d \mathbf{S}, \sigma_z^2) P(a_{k,d} = 1 | \mathbf{a}_{k \setminus d}). \quad (4.7)$$

where \mathbf{z}_d denotes the d th row of \mathbf{Z} and \mathbf{f}_d the d th row of \mathbf{F} .

Second, a Metropolis step [75, 159, 160] is used to propose new features. Assuming fixed means for the prior of \mathbf{W} , the proposal distribution, $q(\theta^+ | \theta)$, for activating K'_+ endmembers for the d th band, is composed of the priors of the latent endmembers and abundances,

$$q(\theta^+ | \theta) = q(\theta^+) = P(K'_+ | -) p(\mathbf{W}) p(\mathbf{S}), \quad (4.8)$$

with $\theta = \{\mathbf{W}, \mathbf{A}, \mathbf{S}\}$ and $\theta^+ = \{\mathbf{W}^+, \mathbf{A}^+, \mathbf{S}^+\}$, where \mathbf{W}^+ , \mathbf{A}^+ , and \mathbf{S}^+ are the proposed additional endmember weights, activations, and their abundances. The acceptance ratio, r_{IBP} , for the Metropolis step is given as

$$r_{\text{IBP}} = \frac{p(\theta^+ | \mathbf{Z}, -) q(\theta | \theta^+)}{p(\theta | \mathbf{Z}, -) q(\theta^+ | \theta)} = \frac{p(\mathbf{Z} | \theta^+, -) p(\theta^+) q(\theta | \theta^+)}{p(\mathbf{Z} | \theta, -) p(\theta) q(\theta^+ | \theta)}.$$

Since the proposal is independent of the previous sample, i.e., $q(\theta | \theta^+) = p(\theta)$, the expression for r_{IBP} can be simplified. Thus, r_{IBP} depends only on the likelihood ratio [159],

$$r_{\text{IBP}} = \frac{p(\mathbf{Z} | \theta^+, -)}{p(\mathbf{Z} | \theta, -)}. \quad (4.9)$$

Sampling from the prior for the feature weights, $p(\mathbf{W})$, is not directly possible and, therefore, a Gibbs sampler is utilized. The new abundances, \mathbf{S}^+ , are sampled from a Gamma distribution with parameters $\frac{1}{K}$ and 1. With these parameters, the mean of the proposal for \mathbf{S}^+ is equal to the mean of the already existing elements in \mathbf{S} . After concatenating the new and existing abundances, the columns of the abundance matrix are normalized to sum to one. This procedure is in line with sampling from a Dirichlet distribution [161].

Algorithm 4.1 Sampling new endmembers using an IBP prior

```

for  $d \in 1, \dots, D$ :
  for  $k \in 1, \dots, K$ :
     $a_{d,k} \sim P(a_{d,k} | -)$ 
   $K'_+ \sim P(K'_+ | -)$ 
  for  $k \in 1, \dots, K'_+$ :
     $a_{d,k}^+ \leftarrow 1$ 
     $\mathbf{w}_k^+ \sim p(\mathbf{W})$ 
    for  $n \in 1, \dots, N_z$ :
       $s_{k,n}^+ \sim \text{Ga}(\frac{1}{K}, 1)$ 
   $r_{\text{accept}} \leftarrow (\text{c.f. Eq. (4.10)})$ 
   $P \sim \mathcal{U}(0, 1)$ 
  if  $\min(1, r_{\text{accept}}) < P$ :
     $\mathbf{S} \leftarrow \text{normalize}([\mathbf{S}; \mathbf{S}^+])$ 
     $\mathbf{W} \leftarrow [\mathbf{W} \ \mathbf{W}^+], \mathbf{A} \leftarrow [\mathbf{A} \ \mathbf{A}^+]$ 

```

As stated in [75], the IBP often suffers from the problem that features are rarely proposed, such that convergence to the stationary distribution is slow. Thus, the authors in [75] propose to augment the acceptance ratio with parameter P_+ , $0 \leq P_+ \leq 1$. This parameter increases the probability of accepting a single new feature, yielding the augmented ratio,

$$r_{\text{IBP, aug}} = r_{\text{IBP}} \cdot \frac{P(K'_+ | -)}{P_+ \mathbf{1}(K'_+, 1) + (1 - P_+) P(K'_+ | -)}, \quad (4.10)$$

with the indicator function $\mathbf{1}(a, b)$ returning one if a and b are equal and zero otherwise. The probability of adding K'_+ features is given in (2.12) in Section 2.3.3.2. Using the

augmented ratio, the probability of proposing new endmembers is increased, leading to faster convergence to the stationary distribution of the Markov chain.

The hyperparameters α_a and β_a of the activation matrix are sampled as described in Section 2.3.3.3. The algorithm for sampling new endmembers is outlined in Alg. 4.1.

4.5.5 Sampling Procedure

We initialize the Gibbs sampler by drawing samples from the prior distributions and start sampling with one feature. The first samples are ignored as several iterations are needed until the Gibbs sampler generates samples from the target distribution.

In contrast to other HSU algorithms, new endmembers are generated during running BNU. Since the newly created endmembers are drawn from the prior distributions, there is the possibility that several endmembers converge to already present ones, increasing the number of endmembers unnecessarily. To solve this problem, we could propose to merge every combination of present endmembers. This would, however, lead to the problem that newly created endmembers are easily removed as they have been sampled from the prior (irrespective of the likelihood) and, thus, they basically present noise.

For this reason, we consider a modified endmember merging strategy, assuming that the probability of merging endmembers is high if the endmembers are similar. Thus, we propose to merge only endmembers that exhibit a correlation above a predefined threshold, T_{Corr} . Using this scheme prevents merging endmembers too early and, additionally, speeds up the process. The endmember fusion proposal is then accepted or rejected in a Metropolis step, similar to reversible-jump Markov Chain Monte Carlo (MCMC) methods [162]. When features shall be merged, elements of the vectors may be active for one feature, but inactive for the other one. For the merged feature, we keep the activation, promoting dense matrices.

Although the Gibbs sampler provides convenient means to sample from a multivariate distribution, it has the disadvantage of sampling incorrectly if the distribution has multiple modes, which are well separated. Especially in cases of many (latent) endmembers, we have observed that the sampler can get stuck in modes. A solution to this problem is the use of Parallel Tempering (PT) [66], where the idea is to run multiple Markov chains in parallel at different temperatures as explained in Section 2.2.3. In our model, the temperature can be understood as an increase of the variance of the

noise such that the likelihood is smoothed. Thus, chains at higher temperatures (higher variances) are likely to jump between the modes of the posterior. The first chain always samples at temperature $T_{PT} = 1$, generating valid samples of the target distribution. PT can be run on a parallel architecture if swaps are proposed every few iterations only. In contrast to standard PT, we cool down the temperatures of all chains, similar as in simulated annealing [163, 164], to ensure that all chains are swapped after several iterations.

An approximation of the MAP estimate of the endmembers, $\hat{\mathbf{F}}$, and the abundances, $\hat{\mathbf{S}}$, is finally given by the sample with the highest posterior probability, which can be calculated from Eq. (4.5). Thus, the obtained estimate is effectively a realization of the variables. This is discussed in Section 4.7.

4.6 Experimental Results

In this section, we evaluate the performance of the proposed approach based on simulated as well as real data. For this, we compare our algorithm, BNU, with different state-of-the-art unmixing algorithms. We consider the following geometrical based methods: Vertex Component Analysis (VCA) [138], Minimum-Volume Enclosing Simplex (MVES) [141], and Sparsity-Promoting ICE (SPICE) [156]. Further, we also provide results for Bayesian Linear Unmixing (BLU) [135].

BNU and SPICE are the only methods that are also able to estimate the number of endmembers, while the other algorithms assume this information to be given. Thus, we compare BNU with the following methods that aim at estimating the number of endmembers: Virtual Dimensionality (VD) [116], Hyperspectral Signal Subspace Identification by Minimum Error (HySime) [153], Denoised Hyperspectral Intrinsic Dimensionality Estimation With Nearest-Neighbor Distance Ratios (HideNN) [155], and SPICE.

For VCA, MVES and BLU, we set the correct number of endmembers. As SPICE and BNU estimate the endmembers, their fractional abundances, and the number of endmembers jointly, we compare BNU especially with SPICE and highlight performance differences between both methods. For BNU, we choose the parameters shown in Tab. 4.1 unless otherwise stated. For the other algorithms, the default values for the parameters are used. Only in case of HideNN, we tuned the parameters as the default values lead to poor results in our simulations. We set the false alarm rate to 10^{-3} for

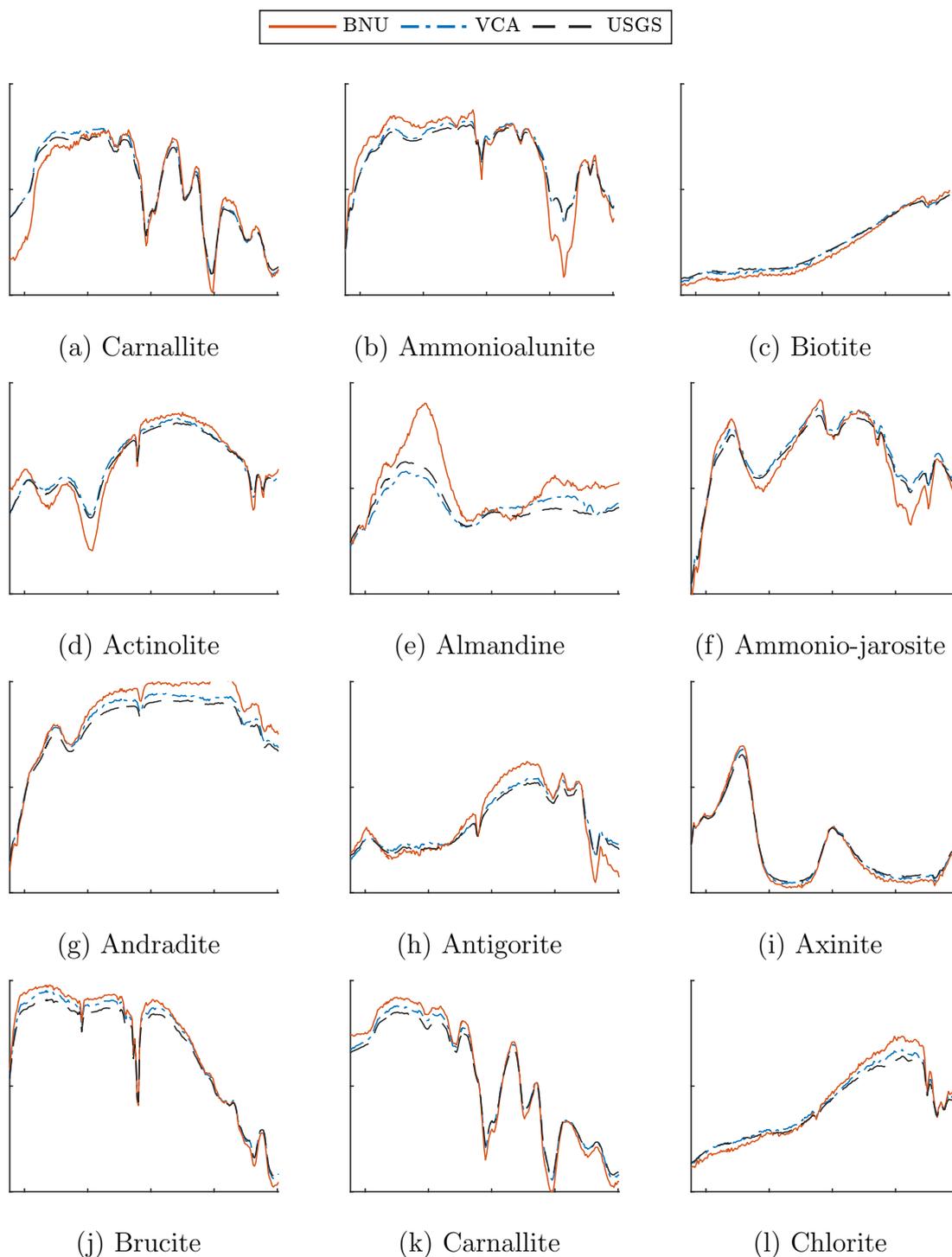


Figure 4.2. Signatures selected from the USGS spectral library [165]. For the simulations, the first K signatures are considered as the endmembers of the simulated hyperspectral image. For comparison, examples of endmembers extracted from a simulated hyperspectral image ($\text{SNR} = 30$ dB) by VCA ($\epsilon_{\text{SID}} = 0.00243$, $\epsilon_{\text{F}} = 1.78$, $\epsilon_{\text{S}} = 23.5349$) and BNU ($\epsilon_{\text{SID}} = 0.0203$, $\epsilon_{\text{F}} = 5.32$, $\epsilon_{\text{S}} = 13.694$) are depicted. The x -axis represents the spectral range from $0.38 \mu\text{m}$ to $2.5 \mu\text{m}$ and the y -axis denotes the normalized reflectance.

Table 4.1. Parameters used for BNU in the simulation experiments

Parameter	Value	Meaning
$h_{\alpha_\sigma}^{(1)}, h_{\alpha_\sigma}^{(2)}$	} 1	hyperparameters for σ_z
$h_{\beta_\sigma}^{(1)}, h_{\beta_\sigma}^{(2)}$		
$h_{\alpha_A}^{(1)}, h_{\alpha_A}^{(2)}$	1	hyperparameters for α_a
$h_{\beta_A}^{(1)}$	1	hyperparameter for β_a
$h_{\beta_A}^{(2)}$	10	hyperparameter for β_a
γ_w	100	weighting of the prior for \mathbf{W}
P^+	0.1	probability of accepting K'_+ features
T_{corr}	0.95	threshold for merging similar endmembers
N_{iter}	10,000	number of iterations of the Gibbs Sampler

VD. Due to the underlying assumptions of HideNN, this algorithm provides floating point estimates of K which are rounded for comparison.

As figure of merits, we consider three different measures: the average angular difference between the true and estimated endmembers, $\check{\mathbf{f}}_k$ and $\hat{\mathbf{f}}_k$ with $k = 1, \dots, K$,

$$\bar{\theta}_{\mathbf{F}} = \frac{1}{K} \sum_{k=1}^K \arccos \left(\frac{\check{\mathbf{f}}_k^T \hat{\mathbf{f}}_k}{\|\check{\mathbf{f}}_k\|_2 \|\hat{\mathbf{f}}_k\|_2} \right),$$

the average angular difference between the true and estimated abundances, $\check{\mathbf{s}}_k$ and $\hat{\mathbf{s}}_k$ with $k = 1, \dots, K$,

$$\bar{\theta}_{\mathbf{S}} = \frac{1}{K} \sum_{k=1}^K \arccos \left(\frac{\check{\mathbf{s}}_k \hat{\mathbf{s}}_k^T}{\|\check{\mathbf{s}}_k\|_2 \|\hat{\mathbf{s}}_k\|_2} \right),$$

and the average Spectral Information Divergence (SID) [166]. The SID measures the difference between the endmembers based on the symmetric Kullback-Leibler divergence [167],

$$\text{SID}_k = \sum_{d=1}^D p_{d,k} \log \left(\frac{p_{d,k}}{q_{d,k}} \right) + \sum_{d=1}^D q_{d,k} \log \left(\frac{q_{d,k}}{p_{d,k}} \right)$$

with $p_{d,k} = \check{f}_{k,d} / \sum_{d'=1}^D \check{f}_{k,d'}$ and $q_{d,k} = \hat{f}_{k,d} / \sum_{d'=1}^D \hat{f}_{k,d'}$. The average SID is given as

$$\overline{\text{SID}} = \frac{1}{K} \sum_{k=1}^K \text{SID}_k. \quad (4.11)$$

Since we perform several Monte Carlo runs, we compute the Root Mean Squared Error (RMSE) of each of the measures, $\epsilon_{\mathbf{F}}$, $\epsilon_{\mathbf{S}}$, and ϵ_{SID} , as in [138].

For the evaluation of the endmember dimension estimation, we consider the average rate of correctly estimated number of endmembers, which we refer to as *Accuracy*. Moreover, to quantify the error introduced by incorrect estimates, we consider the RMSE of the dimension estimates, ϵ_K , over all Monte Carlo runs. However, we want to emphasize that providing a ground truth especially for the number of endmembers in real data is challenging as it strongly depends on the tolerated range of the fractional abundances. In practice, endmembers with very low abundances are neglected.

4.6.1 Simulations

We investigate the effect of the following parameters: (1) the (latent) numbers of endmembers, (2) the noise level, and (3) the strength of multiplicative noise.

For the first experiment, we choose $K \in \{3, 5, 7, 9, 12\}$ pure materials from the USGS spectral library [165]. The chosen endmembers are depicted in Fig. 4.2, along with estimates obtained by BNU and VCA. The ground truth of the abundances is created by drawing samples from a Dirichlet distribution with identical hyperparameters set to $\frac{1}{K}$. With the chosen endmembers and the sampled abundances, hyperspectral images of 40×40 pixels and 224 bands are simulated. Further, Gaussian noise is added, resulting in an SNR of 30 dB.

In the second experiment, the effect of Gaussian noise is investigated by varying the SNR from 10 dB to 30 dB with a step size of 5 dB. The number of endmembers is fixed to $K = 3$.

In the third experiment, the effect of illumination perturbation, i.e., varying lighting conditions, is investigated. Therefore, we apply multiplicative Beta distributed noise to the image with illumination perturbation parameter β_{IP} ($\text{Beta}(\beta_{IP}, 1)$). The higher β_{IP} , the less perturbation is implied, as the probability of sampling values close to one is increased. In contrast, with $\beta_{IP} = 1$, the abundances are scaled by a value drawn from a uniform distribution, ranging from zero to one, leading to highly noisy simulated observations. As in the simulations before, we consider $K = 3$ endmembers and set the SNR to 30 dB.

For each simulation, we present the results over 20 Monte Carlo runs. In low SNR scenarios, K is estimated only in few examples correctly. For this reason, in case of SPICE and BNU, we also include the results when K is overestimated, increasing the number of samples for the calculation of the RMSEs. A significant drawback of HySime and VCA is their requirement of knowledge about the noise. While for HySime, the provided noise estimator is utilized, we provide VCA with the exact SNR.

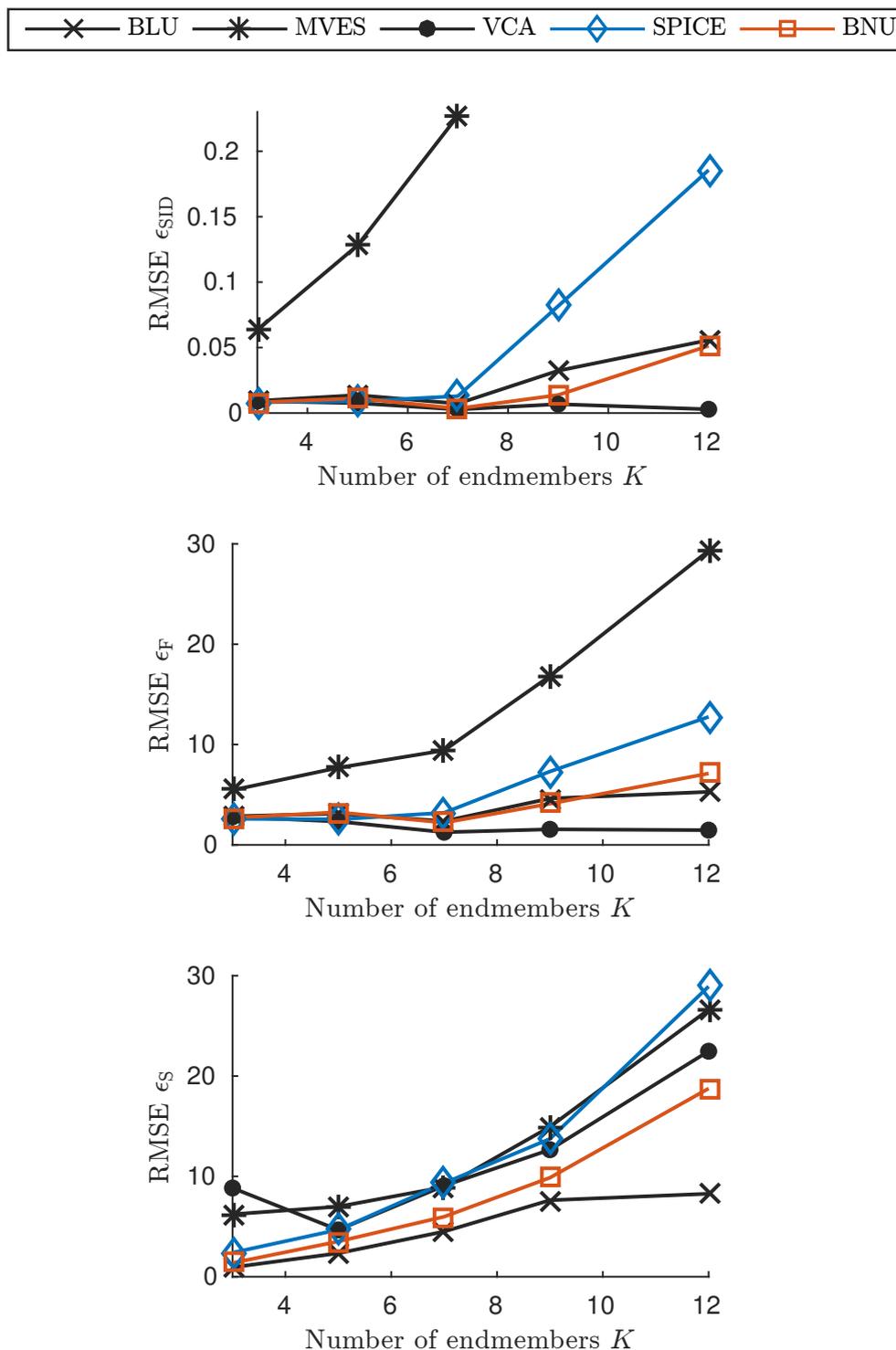


Figure 4.3. Simulation results for different numbers of endmembers from 20 Monte Carlo runs, SNR = 30 dB. All algorithms perform in a similar range, except MVES. BNU clearly outperforms SPICE and provides even for large number of endmembers good results, close to state-of-the-art algorithms. Results with $\epsilon_{SID} > .25$ are not shown for better comparison.

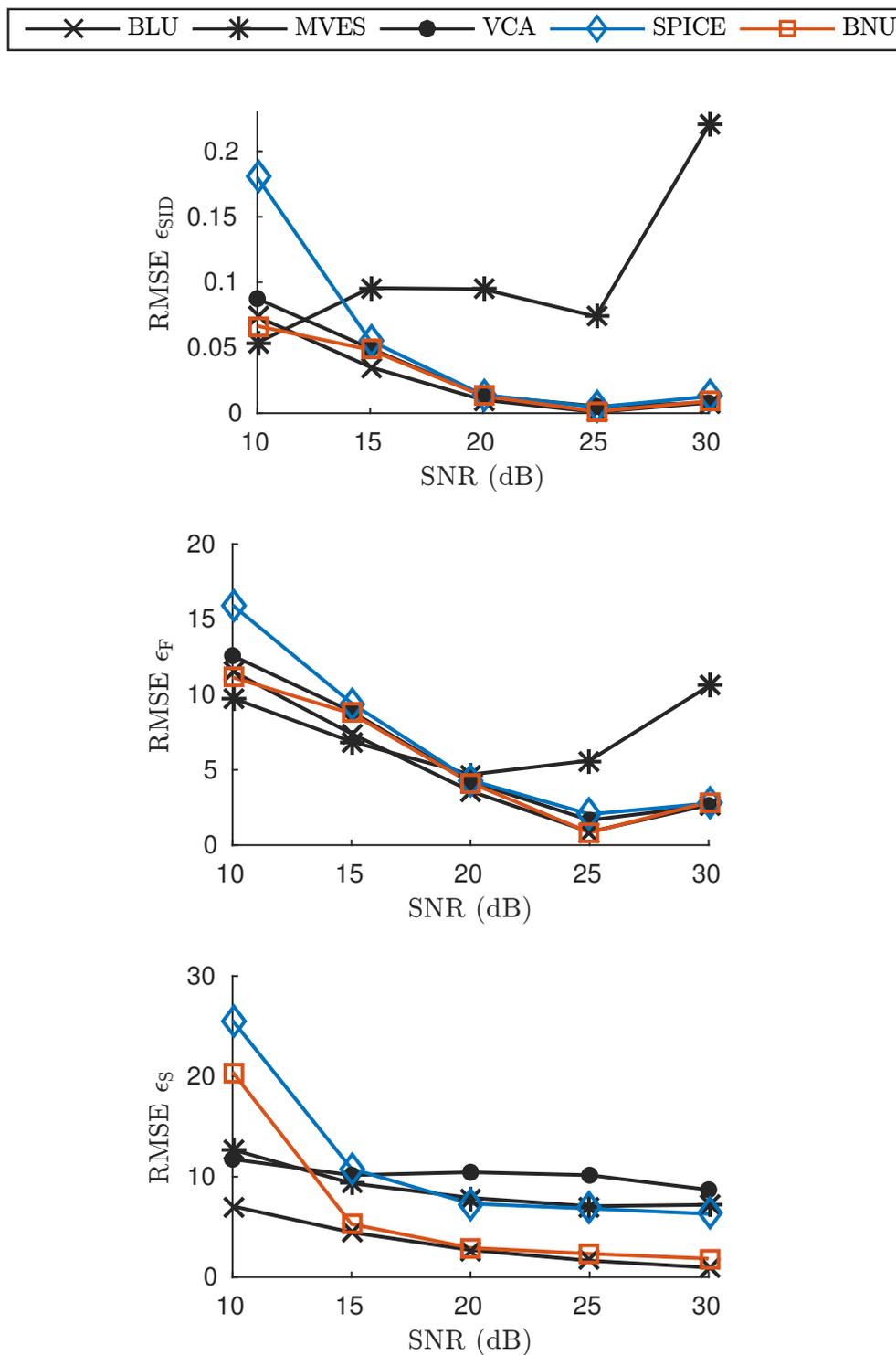


Figure 4.4. Simulation results for different SNRs from 20 Monte Carlo runs, $K = 3$. MVES shows a poor overall performance for endmember extraction, as in some simulation runs it resulted in extremely poor estimates. The other algorithms perform in a similar range, yielding better results with increasing SNR. In all scenarios, BNU yields better results than SPICE, especially in case of low SNR.

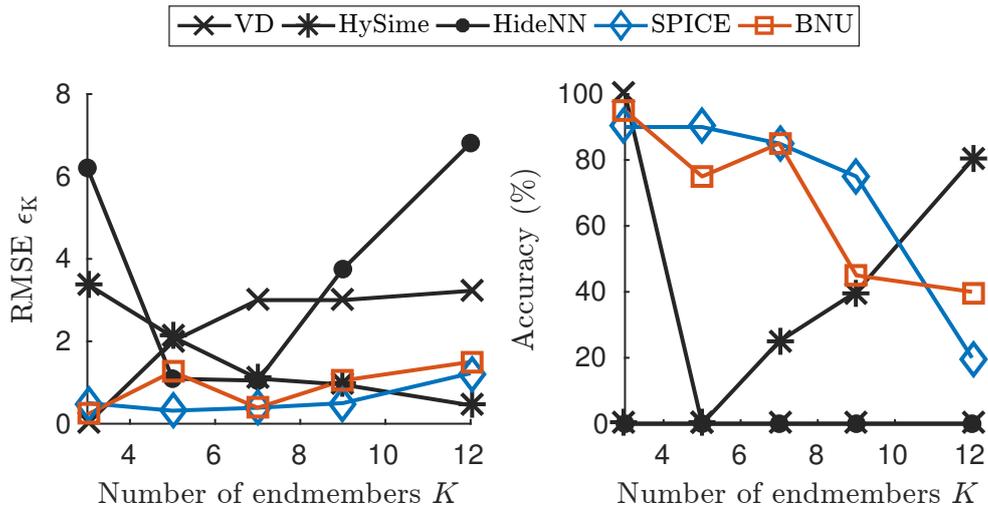


Figure 4.5. Results of the dimension estimation for different number of endmembers, RMSEs (left) and accuracies (right), SNR = 30 dB. BNU and SPICE provide the most accurate estimates. VD and HideNN fail at estimating K correctly for large number of endmembers.

4.6.1.1 Number of Endmembers

The results for different numbers of endmembers are depicted in Fig. 4.3. BNU, BLU and VCA yield the lowest errors in the reconstruction, where BNU clearly outperforms SPICE. In scenarios with many endmembers, BNU provides highly accurate endmember extraction and abundance estimation, exceeded only by BLU. The poor overall performance of MVES can be explained by extremely poor estimates in some simulation configurations, where the other methods still provide good results.

In Fig. 4.2, realizations of the extracted endmembers obtained by BNU and VCA are presented. As shown, BNU and VCA show results of similar high accuracy, while BNU, as opposed to VCA, additionally estimated the number of endmembers.

The results for dimension estimation are shown in Fig. 4.5. BNU and SPICE outperform the other methods, yielding highly accurate estimates. As the number of endmembers increases, the performance of both methods, however decreases. While HySime performs only well for large K , VD, on the contrary, shows good performance only for few endmembers. Despite our attempts to find suitable parameters, HideNN results in the highest error for estimating the number of endmembers. Though showing a low RMSE for $K = 5$ and $K = 7$, it fails to correctly estimate K in any of the simulations.

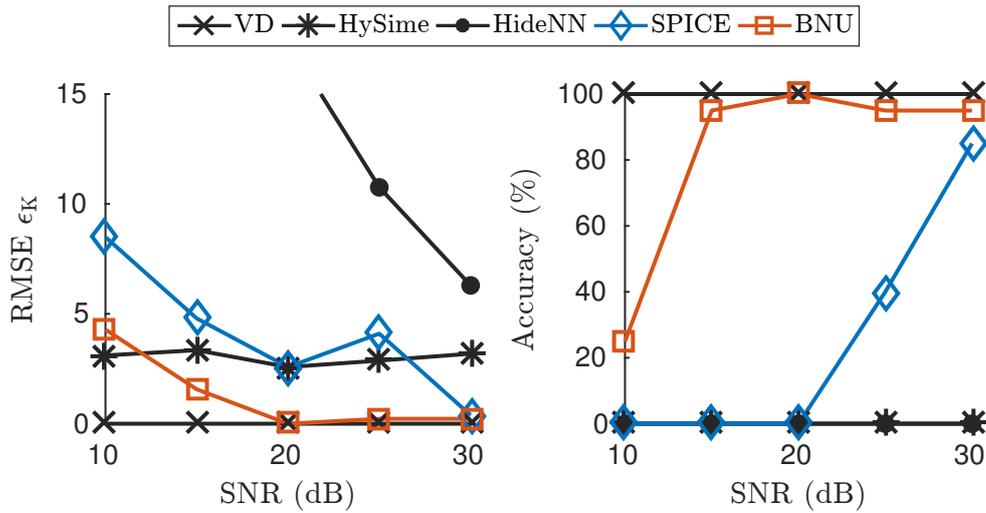


Figure 4.6. Results of the dimension estimation for different SNRs, RMSEs (left) and accuracies (right), $K = 3$. BNU and VCA provide the best results, significantly outperforming the other methods. Note that VCA performs especially well for a low number of endmembers and is likely to show worse results with large numbers. Results with $\epsilon_K > 15$ are not shown for better comparison.

4.6.1.2 Noise Levels

For the evaluation of the effect of noise, different SNR levels have been applied to the simulated image. The results are shown in Fig. 4.4. Concerning endmember extraction, BNU provides highly accurate estimates, similar to the other methods. In comparison to SPICE, BNU clearly results in more precise estimates, especially in the presence of strong noise (SNR < 15 dB).

The abundances are most accurately estimated by BNU and BLU. Only for low SNRs (SNR < 15 dB), BNU is outperformed by BLU. SPICE, however, performs in all scenarios worse than BNU. Although VCA shows generally good results, in practice the results of VCA are likely to be worse since VCA is provided with perfect knowledge of the SNR in our simulations.

The results of the dimension estimation presented in Fig. 4.6 are similar to the results of the previous simulations. BNU and VD provide high accuracies. In case of significant noise (SNR < 20 dB), the accuracy of BNU decreases, but is still remarkably higher than of SPICE. As indicated by the RMSEs, BNU outperforms SPICE for all investigated SNRs. Only VD and HySime are able to provide results similar to BNU in terms of the RMSE.

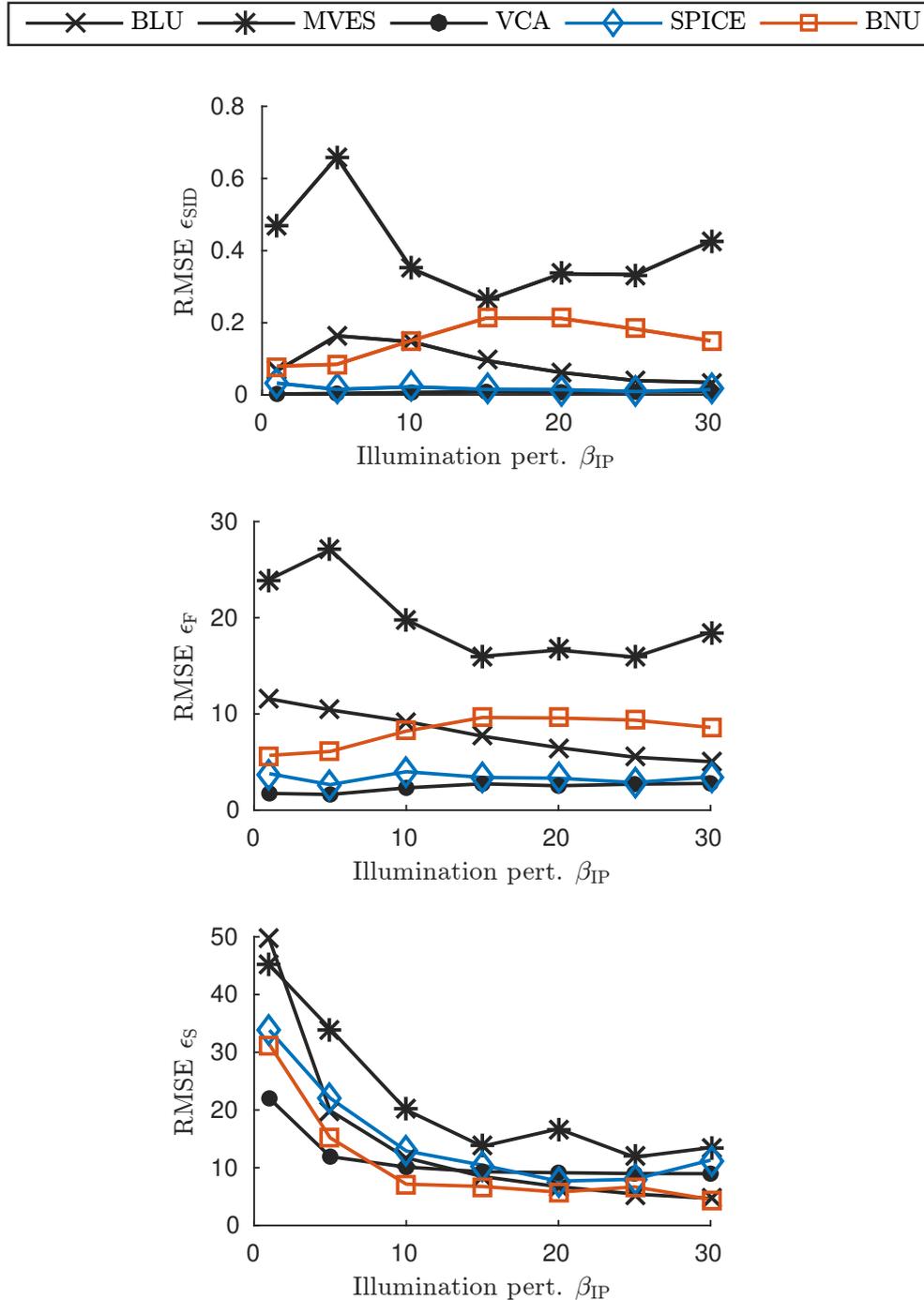


Figure 4.7. Simulation results for different illumination perturbations, SNR = 30 dB and $K = 3$. Low values of β_{IP} lead to low SNRs due to stronger perturbations. All methods except MVES yield good results for endmember extraction. The obtained abundance estimates show strong errors in case of low β_{IP} for all methods. While SPICE outperforms BNU for endmember extraction, BNU shows a better performance for abundance estimation.

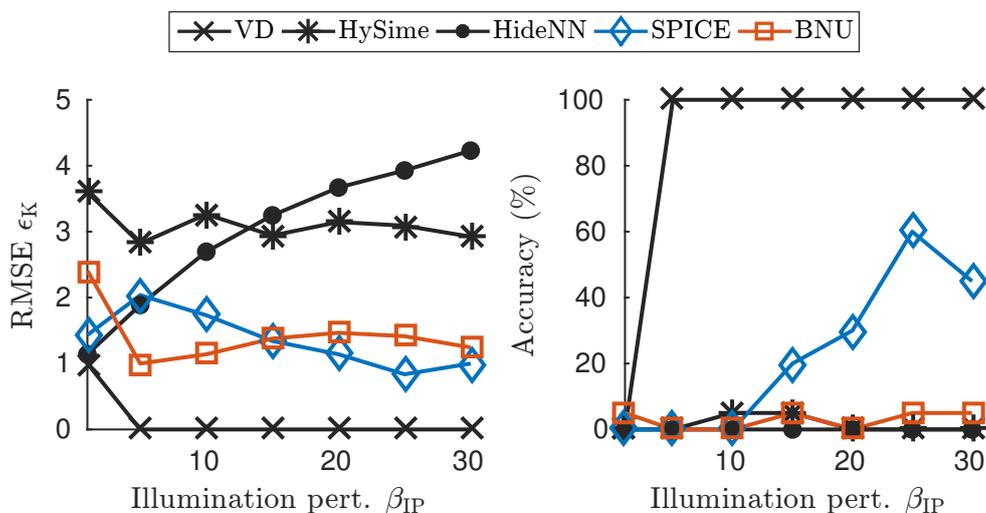


Figure 4.8. Results of the dimension estimation for different illumination perturbations, RMSEs (left) and accuracies (right). Only VD is able to estimate the number of endmembers precisely. BNU provides still good results, overestimating the number of endmembers often by one, yielding similar results as SPICE. The other methods show significant errors.

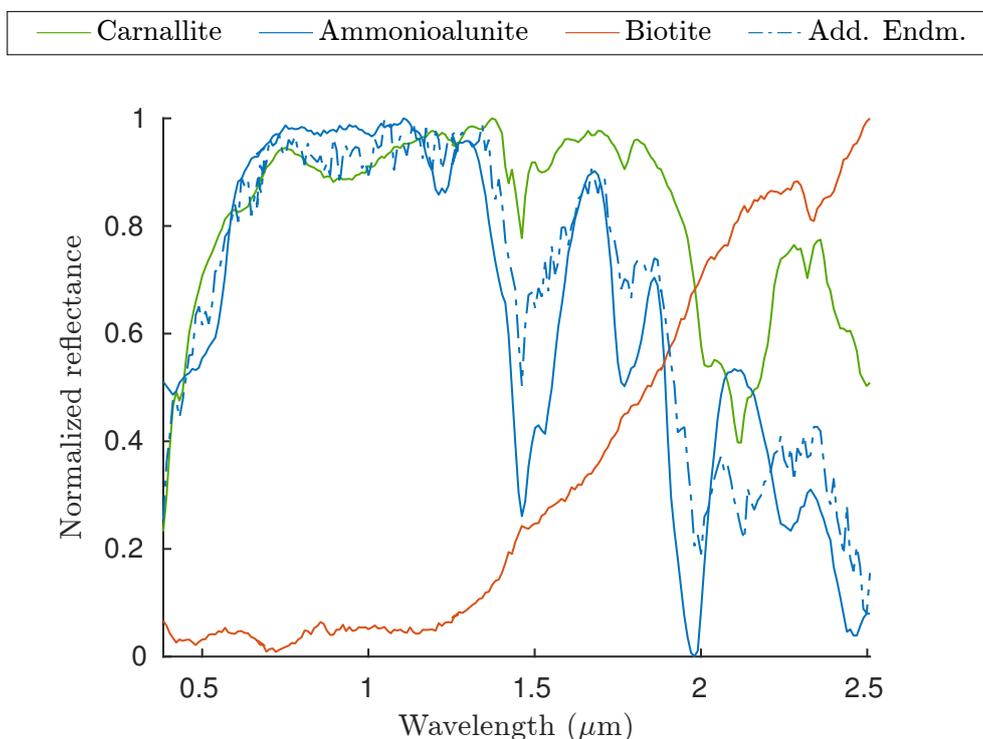


Figure 4.9. Realization of the endmembers for the illumination perturbation simulation, $\beta_{IP} = 25$. The additional endmember is basically a noisy replica of Ammonioalunite, absorbing the multiplicative noise of the observations.

4.6.1.3 Illumination Perturbation

In order to simulate illumination perturbations caused by different lighting conditions, multiplicative noise is applied to the abundances with illumination perturbation parameter β_{IP} as proposed in [138]. Fig. 4.7 reveals that the endmembers are well extracted by all tested algorithms except MVES. BNU and BLU yield similar results, slightly worse than those of VCA and SPICE. However, we need to keep in mind that VCA is provided with the SNR level, in contrast to the other methods. For the estimation of the abundances, BNU provides the most accurate estimates in case of $\beta_{IP} > 5$ and is outperformed only by BLU for stronger perturbations. The drop in the performance of the Bayesian approaches compared with the results of the previous experiments is due to the fact that both generative models do not consider multiplicative noise and, thus, they simply cannot explain the observed noise.

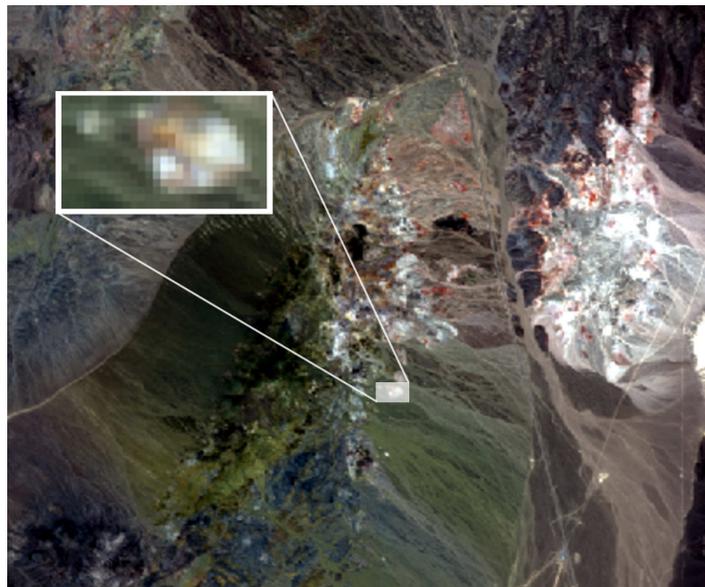
As indicated in Fig. 4.8, none of the examined methods, except VD, is able to estimate the numbers of endmembers in this setup reliably. SPICE recovers the true value, as long as the noise is not too strong. While BNU, HySime, and HideNN mostly fail at estimating the correct dimension, BNU overestimates the number of endmembers by only 1 or 2, indicated by the low RMSE. The additional endmembers often present noisy replica of the endmembers appearing in the scene as illustrated in Fig. 4.9, while the truly present endmembers are well reconstructed. Since BNU does not consider multiplicative noise, the algorithm tries to explain the data by additional endmembers, which we refer to as noise absorption endmembers.

4.6.2 Real Data

For real data evaluation, we consider subsets of (i) the Moffett field and (ii) the Cuprite scene. The Moffett field contains vegetation and urban features. In contrast, the Cuprite scene shows mainly geological signatures of different minerals and rocks. For comparison, we show the endmembers estimated by VCA and the most similar endmembers chosen from the ASTER spectral library [168]. We choose VCA for comparison as VCA provides highly accurate estimates and is one of the most well-known unmixing methods. To obtain an approximate MAP estimate for BNU, the sample maximizing the posterior is chosen after 10000 samples, though BNU converged to the stationary distribution already after a few hundred iterations. Five Monte Carlo runs are performed to make sure that the results are independent of the (random) initialization. For the Moffett field, we set $\gamma_w = 2000$ and for Cuprite $\gamma_w = 500$. Since no ground truth is available, for better and detailed comparison, we use a subset of both data sets as in [135, 149]. Further, the SNR is assumed to be 30 dB for VCA.



(a)



(b)

Figure 4.10. True color image of the (a) Moffett field and (b) Cuprite scene. The selected subset of the Moffett field has a size of 50×50 pixels and contains different signatures such as of soil, vegetation, and water. The selected subset of the Cuprite scene has a size of 16×28 pixels. The subset contains signatures from various minerals.

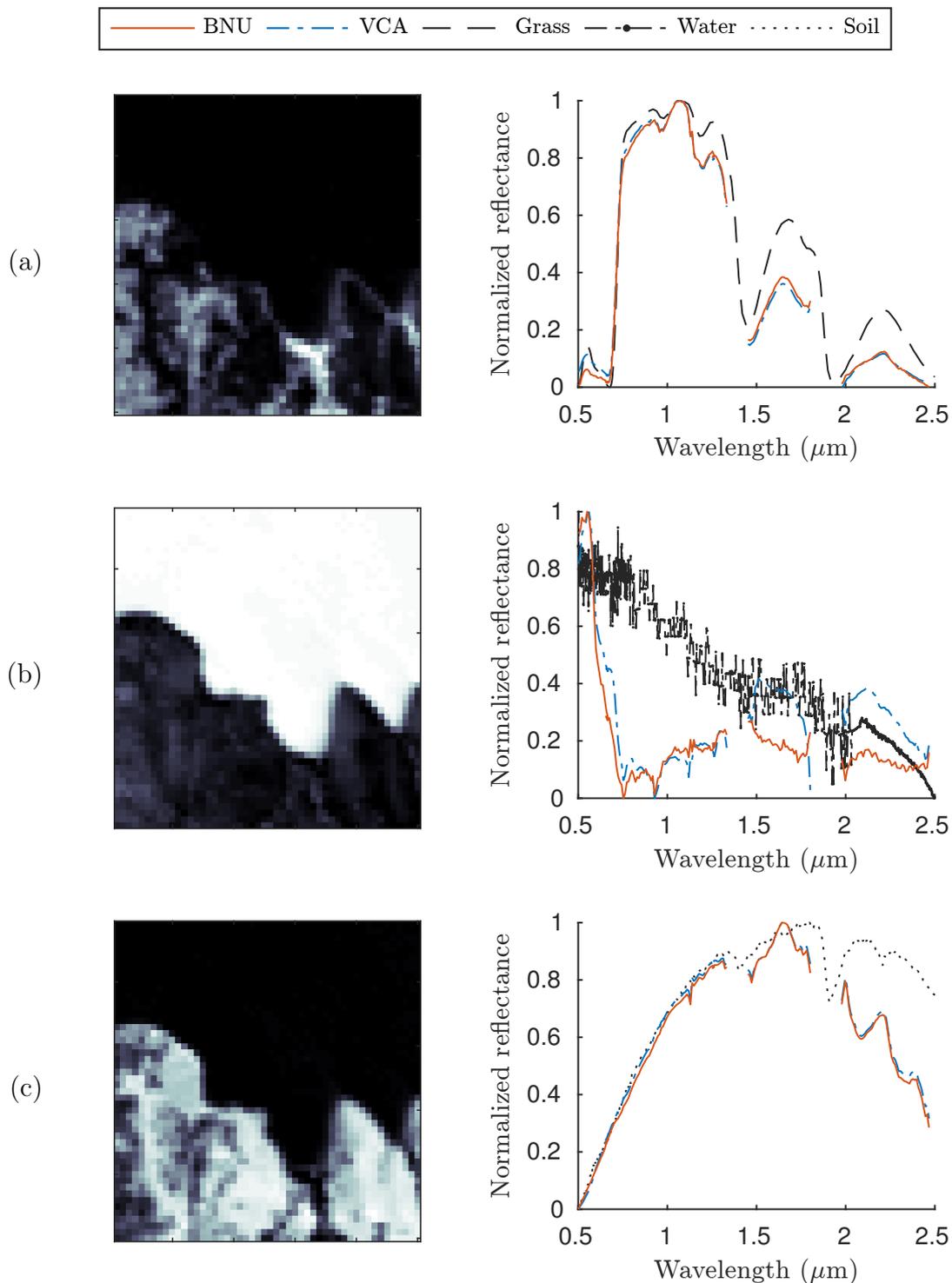


Figure 4.11. Results of the Moffett field, estimated abundances (left) and the extracted endmembers (right). For the abundances, *white* refers to 1 and *black* to 0. In each of the five Monte Carlo runs, BNU extracted three endmembers: (a) grass/vegetation, (b) water, and (c) soil. The estimated abundances represent the scene well and are similar to the results shown in [135].

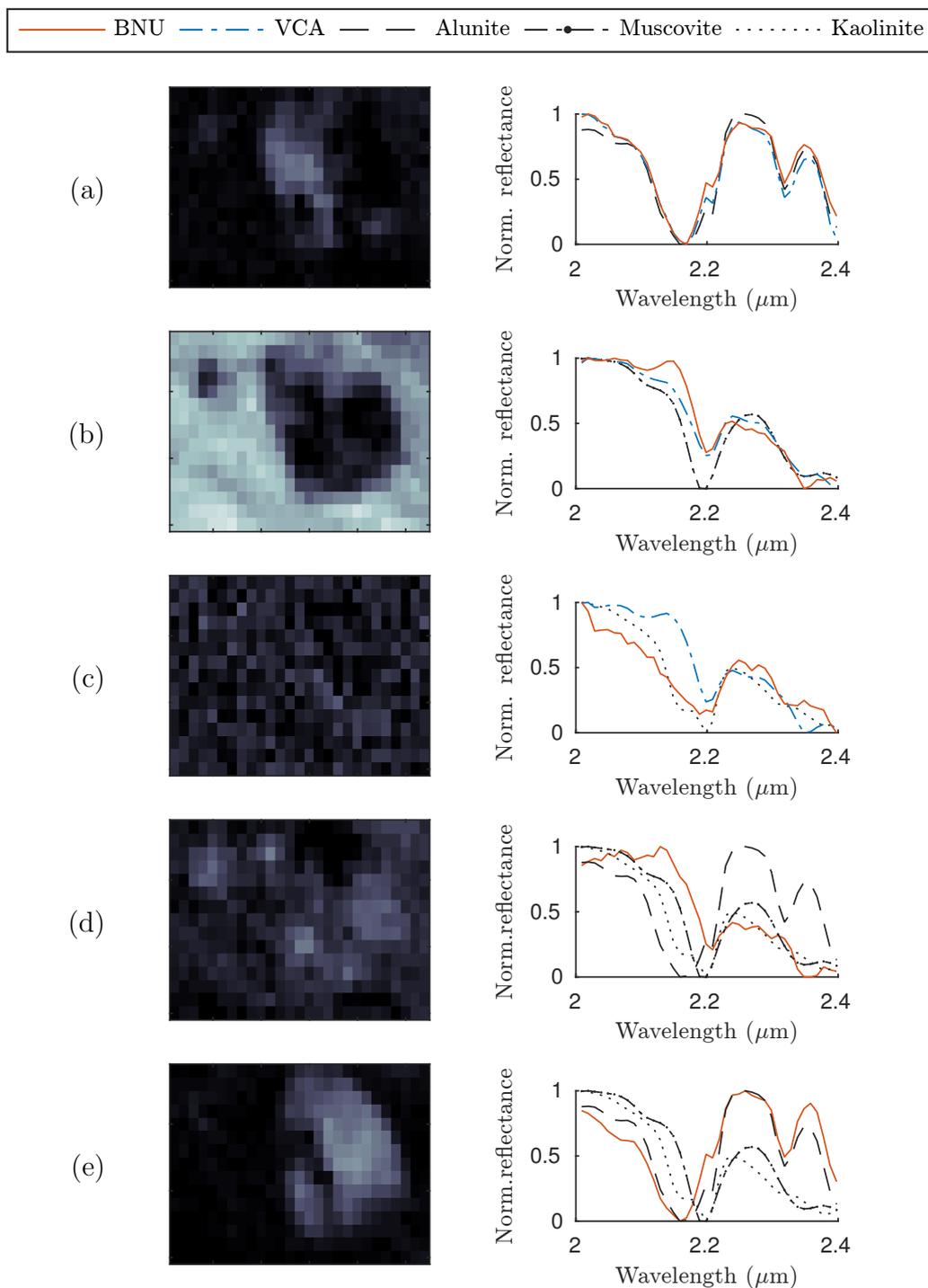


Figure 4.12. Results of the Cuprite scene, estimated abundances (left) and the extracted endmembers (right). For the abundances, *white* refers to 1 and *black* to 0. BNU samples between four to five endmembers in all Monte Carlo runs. In the illustrated realization, (a) Alunite, (b) Muscovite, (c) Kaolinite, and noisy copies of (d) Kaolinite and (e) Alunite are extracted. Though in this example, the feature merging proposal is rejected, the results are similar with those reported in [135].

4.6.2.1 Moffett Field

The Moffett field scene was captured over Moffett Field (CA, USA) by AVIRIS in 1997 and has been explored in many studies, e.g., [135, 169, 170]. The scene is represented in 224 bands, covering the spectral range from 400 nm to 2500 nm with a spectral resolution of $20\text{ m} \times 20\text{ m}$ per pixel. After removing the noisy bands, 188 bands are left for the evaluation. We follow [135, 149] and select a subset of 50×50 pixels size, which is depicted in Fig. 4.10(a).

As shown in Fig. 4.11, BNU and VCA extract grass, soil, and water signatures as endmembers. The estimated abundances show the distribution of water, soil and grass, as typical for a coastal scenery. This is in line with the results found in [135]. VCA and BNU provide results of similar accuracy, with BNU correctly estimating three endmembers in each of the five Monte Carlo runs.

4.6.2.2 Cuprite Scene

The Cuprite scene was also captured by the AVIRIS instrument in 1997. Since then it has been extensively studied in HSI research. The scene covers the Cuprite mining area in Nevada, USA, containing different minerals mainly with a spatial resolution of $20\text{ m} \times 20\text{ m}$. The subset depicted in Fig. 4.10(b) of $28\text{ pixels} \times 16\text{ pixels}$ size shows mainly three different materials, as detailed in [135]. Although some bands suffer from strong noise, noisy bands are not removed in this data set.

The endmembers in this scene are strongly correlated, in contrast to those of the Moffett field, rendering extraction more challenging. After convergence to the stationary distribution, the Gibbs sampler in BNU draws samples with four to five endmembers. A sample with five endmembers is shown in Fig. 4.12, where the spectral range is limited between $2\text{ }\mu\text{m}$ and $2.4\text{ }\mu\text{m}$ for detailed analysis. BNU results in accurate reconstructions of the endmembers, slightly better than VCA, especially for Kaolinite. The additionally extracted endmembers are similar to Kaolinite and Alunite and can be considered as noise absorption endmembers.

4.7 Discussion and Outlook

As demonstrated on simulated as well as real data, BNU is able to accurately extract the endmembers and estimate the abundances, providing results comparable with state-of-the-art algorithms. BNU additionally infers the number of endmembers from the

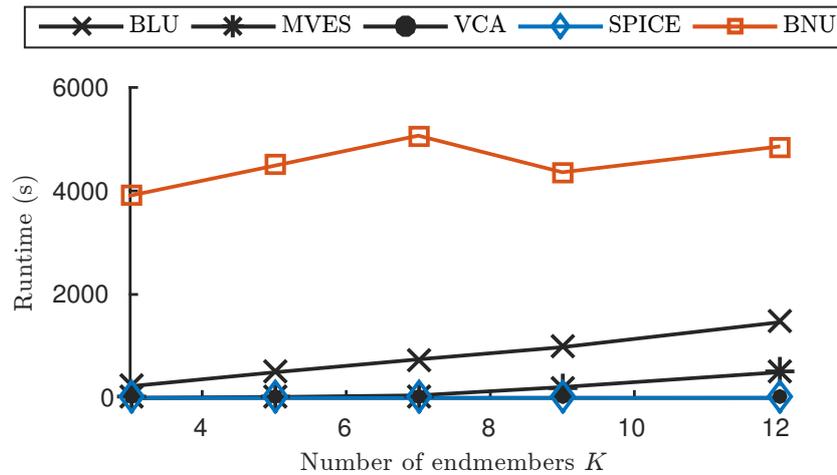


Figure 4.13. Runtime comparison of the algorithms. Due to parallel tempering (5 Monte Carlo chains), BNU is the slowest approach, significantly slower than the other approaches. Please note that the algorithms have not been optimized.

observations, while most existing approaches assume this number to be known. SPICE is also able to estimate this information jointly, but often results in less accurate estimates in comparison with BNU, especially in the presence of strong noise.

One drawback of BNU is, as common for algorithms based on Bayesian inference, the computational complexity of the algorithm. As an example, Fig. 4.13 compares the runtime of the algorithms during the simulation of different numbers of endmembers. Note that the algorithms are not optimized in terms of speed. For BNU, five Markov chains are run for PT on a single core architecture. We set the number of samples drawn by the Gibbs sampler to 1000 for BNU, after which no improvements of the MAP were found. In contrast to BNU, BLU usually requires far less samples and is, therefore, much faster thanks to a better initialization by means of an (arbitrary) unmixing algorithm.

In order to reduce the runtime of BNU, a variational approach of the IBP has been proposed in [160]. Depending on the application, the authors in [160] found that the approximations introduced by the variational algorithm can lead to inaccurate and slow inference. In some cases, the Gibbs sampler is not only more accurate, but also faster. Further, the authors reported that the variational approach is useful when the number of variables was sufficiently large. In [171], it is shown that the IBP is related to Beta processes. This similarity could be exploited to derive new variational methods based on existing approaches for the Beta processes, e.g., in [172].

Since BNU is based on Bayesian inference, it tries to explain the observed data and expresses its belief over the unknowns by means of the posterior. As we have seen in the

experiments in Section 4.6.1.3 and Section 4.6.2.2, noisy observations can have a two-fold effect on the inference: either the variance of the noise is increased or the number of endmembers is overestimated. In the latter case, the noise is basically absorbed into the additional endmembers, which happens mostly if the observations suffer from noise which cannot be explained by the model. Consequently, in the case of strong noise, there often exist several, equal probable explanations for the observations. The MAP estimator, however, considers only one explanation as the estimate consists of the most probable sample only (Section 4.5.5). Thus, the MAP estimate contains only limited information about the posterior. Better generalization capabilities are provided by estimators that take the shape of the posterior into account, e.g., the Minimum Mean Squared Error (MMSE) estimator. The MMSE estimator, however, cannot be used with the IBP to infer the endmembers due to the varying dimensions of the samples. Hence, the problem remains to find better estimators for the use in BNU.

Another issue concerns the choice of the hyperparameter γ_w . The prior for the feature weights favors similar bands (and hence similar endmembers), where γ_w controls the tolerated variance. A high value of γ_w results in similar endmembers such that newly introduced endmembers are likely to be rejected or merged with existing endmembers. An important future direction is the development of a suitable prior for the endmembers with known normalization, such that the hyperparameters can be sampled, and thus, be learned from the observed data. An alternative to finding a different prior is Meta-learning [173]. Given a database of many data sets with known suitable parameters, meta-learning finds data sets in the data base that are similar to the given one and uses the known parameters to propose parameter settings for the data set at hand.

4.8 Conclusion

In this chapter, we have considered Hyperspectral Unmixing as a feature learning problem. In particular, we have utilized the BNFL framework, in which we incorporated the assumptions on the endmembers and abundances. A remarkable advantage in comparison to existing unmixing algorithm is that the proposed method allows to infer the abundances, the endmembers and the number thereof jointly. We propose the use of PT to alleviate the problems induced by the multi-modal posterior distribution. The simulations experiments show that the desired quantities can be reliably inferred. In case of low SNR, the proposed algorithm tends to overestimate the number of endmember slightly, producing noisy replica of present endmembers. Applying the algorithm to real hyperspectral images results in accurate estimates with results that are in line with those reported in previous work.

Chapter 5

Bayesian Feature-Based Learning From Demonstrations

In this chapter, we extend the BNFL framework for the use in decision-making. We assume that the states of the agent are composed of several latent features, which determine the action the agent takes. The benefits of this approach are two-fold: first, the states can be represented compactly, rendering the decision-making problem much more efficient as compared to working in the original domain. Second, the features can be regarded as causes for the observed decisions, allowing for a deeper understanding of the observed behavior.

This chapter is organized as follows. In Section 5.1, we introduce and motivate concepts for Learning From Demonstrations (LFD). We highlight the contributions of this chapter in Section 5.2 and formulate the problem in Section 5.3. Afterwards, we give an overview of the state of the art of feature learning in decision-making in Section 5.4. In Section 5.5, we motivate and present the model for feature-based decision-making. A Bayesian model is then detailed in Section 5.6 and we explain how the number of latent features can be inferred from the data. Section 5.7 explains the inference scheme for the latent variables. The proposed algorithm is empirically evaluated with simulation experiments in Section 5.8, demonstrating the performance of the algorithm. Real data experiments are conducted in Section 5.9. We discuss our findings in Section 5.10 and end with a short conclusion in Section 5.11.¹

5.1 Motivation and Introduction

Decision-making plays a crucial role in many applications, such as robot learning, driver assistance systems, and recommender systems [174]. A fundamental question in decision-making is how an agent can learn to make optimal decisions. Learning from an experienced teacher provides a natural means to solve this problem, without the need of explicitly defining rules for the desired behavior. Further, this approach naturally avoids risky states, a significant problem in Reinforcement Learning (RL)

¹This chapter has served as basis for the journal article: J. Hahn and A. M. Zoubir, “Bayesian Nonparametric Feature and Policy Learning for Decision-Making”, submitted to *Pattern Recognition*, 2016.

approaches that we investigate in [175]. Further, observing a teacher may provide a deeper understanding of the decision-making process. Therefore, LFD [4] has gained a lot of interest in the recent past.

According to [4], approaches for LFD can be grouped into (i) reward-based models and (ii) imitation learning. In reward-based models, it is assumed that the agent makes its decision based on a reward which is, in the context of LFD, learned from observations (as in Inverse Reinforcement Learning (IRL) [176]). Inferring the reward can be challenging and involves solving a MDP, usually by means of a RL algorithm. We proposed an Expectation Maximization framework for IRL in [177]. In imitation learning, it is assumed that an experienced teacher can be observed. Thus, the policy, telling the agent how to act in a given situation, can be learned by understanding the direct relation between the teacher's states and actions. Especially in reward-based approaches, problems defined on infinite spaces, where the state of an agent can take continuous values, are mostly intractable to solve and efficient approximations are needed. Only in the case of finite state and action spaces and known rewards, learning optimal policies has been solved [178]. A typical approach to facilitate this problem is the representation of the state space in a feature domain, c.f. [179, 180].

However, decision-making has been viewed only from a limited feature-based perspective, where features are usually designed by experts and mainly serve the purpose of reducing the size of the state space. We argue that, in many practical systems, the agent makes its decision based on a compact representation of the observed data, which can be considered as a projection onto a feature space of the decision-making problem. As an example, consider a person driving a car. The driver's observations consists of, i.a., the location, speed, and acceleration of his vehicle and the surrounding vehicles, the type of the road and the weather conditions. However, the driver makes his decision based on a subset of the available information, e.g., on the time-to-reach between his and the other road users' cars. This idea has also been investigated from a psychological point of view by the concept of discovering latent causes in human behavior, which is related to learning state space representations [181].

5.2 Contributions

Assuming that a certain structure underlies the observations, we aim at inferring the latent features and build a feature-based representation of the states, yielding the following two advantages: first, the states can be represented compactly, rendering the decision-making problem much more efficient as compared to working in the original

domain. Second, the features can be regarded as causes for the observed decisions, allowing for a deeper understanding of the observed behavior. In particular, we consider a LFD problem and propose a Bayesian nonparametric framework for feature learning in LFD. We assume that the policies depend on the features of the observed demonstrations. With this model, we are able to (i) significantly reduce the state space by (ii) learning the features as well as the number of features and (iii) provide a better understanding of what caused the teacher to take the observed actions.

5.3 Problem Formulation

The goal of this work is to introduce a feature-based LFD framework. While this model can be used for teaching an agent given observations of the desired behavior, the focus of this work lies on the analysis of the observed behavior by means of the features. Since we consider a decision-making task, the investigated problem can be modeled by means of a Partially Observable Markov Decision Process (POMDP) [182, 183], which is defined by

- a set of observations, \mathcal{O}_z ,
- a set of states, \mathcal{O}_s ,
- a finite set of N_u actions, \mathcal{O}_u ,
- a transition model, which describes the probability of entering a state after taking an action in the current state,
- an observation model which explains how the observations are generated from the states,
- a discount factor, which penalizes long-term rewards,
- and a reward function, R .

As the main goal of this work is to provide a means to understand observed behavior, we consider an imitation learning approach and learn the relation between states and actions from the observations directly, where the reward and, hence, the discount factor, are not considered.

We assume that we (or an agent) have access to N_z noisy observations, $\mathbf{z}_n \in \mathcal{O}_z$, of the states, $\mathbf{x}_n \in \mathcal{O}_s$, with $n = 1, \dots, N_z$. In particular, we consider Gaussian

noise, i.e., \mathbf{z}_n , describes observations of the states, \mathbf{x}_n , with additive Gaussian noise. Further, we assume that the actions, $u_n \in \mathcal{U}$, taken in the corresponding states, can be observed. The observations are assumed to be optimal in the sense that they represent the behavior of the agent seeking its goal, i.e., without any exploratory steps.

A simple approach to the considered problem would be the use of a feature extraction technique such as PCA [9, 10] or NMF [132] to learn features from the observed states.

Following this solution, features cannot be jointly learned and shared among different actions. Moreover, learning discriminative features is not guaranteed. Therefore, we argue that the features and policies need to be learned jointly such that a trade-off between feature sharing, promoting a compact model, and discrimination capability is found based on the observations.

5.4 State of the Art

In the following, we provide an overview of the current state of the art of feature representations in LFD. As IRL includes the problem of solving a RL problem, we start this section by first giving an overview of feature learning for RL.

5.4.1 Reinforcement Learning

Large state and action spaces are especially problematic for value-based RL algorithms such as value-iteration [178, 184] or Q-learning [185] since the value function, representing the expected reward for each state, needs to be approximated. In early approaches, a pre-defined set of basis functions is linearly weighted to represent the value function. These basis functions are often referred to as feature mappings. Exploiting the linear relationship, efficient methods for estimating the values are proposed e.g., in [180]. However, only few approaches exist to learn the basis functions. For example, in [186], a feature selection approach is presented. By means of an l_1 -regularized approximate linear program (RALP) the value function is approximated by selecting features from a large set of potential features.

To learn nonlinear features, Riedmiller [187] has proposed the Q-fitted value iteration. Here, the value function is approximated by means of a neural network and the features are learned in the hidden layers of the network. Although this approach shows good

performance in practice, it lacks convergence guarantees. In [43], the neural network is replaced with a deep-layered counterpart. The capability of this architecture is demonstrated on playing old computer games, often outperforming experienced human players. Research in this area is ongoing. The latest results have gained global public interest in a competition of the traditional Chinese game *go* between a human expert and an artificial intelligence [44]. However, the underlying deep architectures needs to be designed carefully and parameter tuning can be challenging. As the inferred features mainly serve the purpose of dimensionality reduction, they do not necessarily possess a meaning that can be easily interpreted. A survey on feature learning for batch-based reinforcement learning is provided in [188].

A different concept of features is proposed by Hutter with the Feature Reinforcement Learning framework [189]. The goal is to learn a feature mapping from the agents history (comprised of actions, states, and rewards) to an MDP state, enabling decision learning for infinite state spaces [190].

The Contingent Feature Analysis (CFA) [40] is motivated by the question, which information is needed to understand the behavior of an agent. For this, features are sought for that explain the high variance of the temporal derivative of the observed states, when the agent performs an action.

An alternative framework for learning the latent structure in the state space based on a [191] is proposed in [192]. In this framework, it is assumed that the observed states can be compactly represented by exploiting the structure within the states, enabling efficient learning. An extension to an online approach has been proposed in [193], where the features are selected from a large set by means of Group LASSO [194].

5.4.2 Inverse Reinforcement Learning

IRL is concerned with the problem of learning the reward function from observed behavior [176]. In the context of IRL, features are mainly used to parameterize the reward function, often in a linear way, e.g., in [176, 177]. Recent attempts have been made to consider a DL architecture [195] for feature learning.

A Bayesian nonparametric approach is proposed in [196]. Here, an IBP prior is utilized to model feature activations. As the features of the reward function are assumed to be known, this approach can be understood rather as a feature selection than feature learning for IRL, where the number of features is estimated by the IBP. Different results

on Bayesian nonparametrics for IRL, that is indirectly related to feature learning, are given in [197], where a partitioning of the state space is sought for, or [198], where complex behavior is decomposed into several, simpler behaviors that can be easily learned.

5.4.3 Imitation Learning

Instead of estimating the reward as in IRL, Imitation Learning aims at inferring the underlying policy directly. Usually, handcrafted features are used, e.g., in [199–201]. Attempts to introduce new features are made in [202] as an extension of the maximum margin planning algorithm proposed in [199]. As explained in [4], imitation learning can be considered as a supervised learning task. Thus, feature selection and learning techniques developed for classification and regression can also be used in imitation learning. An excellent overview is given in [14]. Though these models work well in practice, they might not be able to provide a deeper understanding of the observed behavior as they do not explicitly model the states and policies.

5.5 Choice of the Model

In the first part of this section, we propose a feature model for LFD. In the second part, we explain the relevance of the transition model to the proposed framework. Since we assume a mixture of policies in this framework, we briefly discuss the intuition behind this assumption in the third part. Alternative models for feature learning for LFD are discussed in the forth part.

5.5.1 Feature Model for Learning From Demonstrations

We assume a linear latent feature model, similar to NMF [132] and PCA [9,10]. Thus, the noisy observations, $\mathbf{z}_n \in \mathbb{R}^{D \times 1}$, $n = 1, \dots, N_z$, are assumed to be composed of the latent features, $\mathbf{F} \in \mathcal{F}^{D \times K}$, and the feature coefficients, $\mathbf{s}_n \in \mathcal{S}^{K \times 1}$,

$$\mathbf{z}_n = \mathbf{F}\mathbf{s}_n + \boldsymbol{\epsilon}_n, \quad (5.1)$$

where $\boldsymbol{\epsilon}_n$ represents Gaussian i.i.d. noise with variance σ_z^2 and the states are given as $\mathbf{x}_n = \mathbf{F}\mathbf{s}_n$. The number of features is K and the dimension of the observations is D .

Clearly, the feature space, \mathcal{F} , depends on the application. In the following, we assume that the features are positive-valued, i.e., $\mathcal{F} = \mathbb{R}_+$. Following [75], the feature matrix is composed of a binary activation matrix, $\mathbf{A} \in \{0, 1\}^{D \times K}$, and a weighting matrix, $\mathbf{W} \in \mathcal{F}^{K \times D}$, where the relation is given by the Hadamard product,

$$\mathbf{F} = \mathbf{A} \odot \mathbf{W}. \quad (5.2)$$

The feature model in Eq. (5.2) can be easily extended to an infinite feature model by placing an IBP [15, 79] prior over \mathbf{A} as explained in Section 2.3.3. The IBP assumes an infinite number of features, while the observed data can be explained by a finite number. This gives rise to a nonparametric model, where the number of features is implicitly modeled by means of the IBP. This is detailed in Section 5.5.

A fundamental difference to most existing work on decision-making using feature representations is that we assume that the agent makes its decision based on features, where each feature attracts the agent to take a specific action. We consider a (latent) linear feature model where the feature coefficients depend on the observed state and determine the actions. For this reason, we refer to the feature coefficients also as *substates*.

5.5.2 Transition Model

In decision-making problems, the transition model explains which actions the agent can take in each state. Thus, the transition model acts as a constraint on the possible actions. As we focus on inferring the latent causes for the observed actions by means of features, the transition model is less relevant as it only provides additional information. Besides, the transition model is rarely known in practice. As we are given observations, we can, theoretically, infer the transition model. For this, we could either employ a parametric or a nonparametric model. Defining a parametric model for the transitions is not trivial, as the dynamics in the latent space may be highly nonlinear. Alternatively, a nonparametric model can be assumed. This, however, requires a large amount of observations for the estimation of the parameters, which is often not available. Assuming that the noise in the observations is low, we argue that we can reliably infer the substates from the corresponding observations, eliminating the need for a transition model.

5.5.3 Feature-Based Policy

As each feature imposes its own policy, the probability of the agent taking an action, u , in a substate, \mathbf{s} , is a mixture of the feature policies, $P(u | \phi_k)$,

$$P(u | \mathbf{s}, \Phi) \propto \sum_{k=1}^K s_k P(u | \phi_k), \quad (5.3)$$

where ϕ_k , $k = 1, \dots, K$, are the parameters of the feature policies and $\Phi = [\phi_1, \dots, \phi_K]$. The mixture of policies can be interpreted either as a stochastic or a deterministic policy. In the first case, the action to be taken should be sampled according to Eq. (5.3). In the second case, simply the most probable action is taken. Mixture policies have been investigated in multi-objective problems, where an agent aims at reaching several objectives, some of which can even be conflicting [203–205]. A stochastic policy is needed in this framework to ensure that all goals can be satisfied. Similar problems occur if the problem at hand is described by a POMDP, where the true state of the agent is unknown. The uncertainty about the state of the agent is expressed by beliefs over states. Acting according to a stochastic policy maximizes the expected return [182].

In case of single agents, it has been shown that deterministic policies are optimal solutions for MDPs [206]. As explained in Section 5.5.1, we argue that we can reliably infer the substates. Thus, we assume a deterministic policy in the following, where the probability in Eq. (5.3) expresses our confidence about the actions given the states.

Note that we also assume that each feature imposes a deterministic policy such that we expect a strongly peaked distribution for the feature policies, expressing the confidence of the chosen action.

5.5.4 Alternative Feature-Based Models

Of course, there are other possibilities to incorporate feature learning in LFD. We briefly discuss two alternatives with their potential advantages and drawbacks. For completeness, as mentioned in the introduction, in the most trivial setup, one could simply cluster the states according to the observed actions and then learn the features. This, however, has the significant drawback that the features cannot be shared among the clusters.

5.5.4.1 Unique Coefficient Model

Instead of assuming, as in our model, that the features determine the behavior of the agent, the feature coefficients can be clustered, where the clusters indicate the optimal actions given the coefficients. Thus, the clusters can be interpreted as latent substates. Supposing that the elements of the feature coefficients are binary-valued, we can convert each feature coefficient vector to a unique identifier, representing the cluster. Thus, instead of employing costly clustering, a fast deterministic mapping from the binary coefficients to the cluster identifier can be used. However, as the identifiers must be unique for the cluster assignments, we can have at maximum 2^K different clusters in this setup, i.e., the number of clusters (and, hence, possible actions) is strictly limited by the number of features. An advantage of this model is that the relation between substates and policies can be nonlinear. However, a significant drawback is that this model suffers severely from errors in the inference of the substates. Consider the case, where, for instance, due to noise, one element of the substate is incorrectly set. This substate is then assigned to a new cluster, for which the policy must be inferred from potentially little data, yielding highly varying policy estimates. As in our approach, an IBP prior can be placed over the feature activations to infer the number of latent features.

5.5.4.2 Clustering-based Approach

A clustering-based approach assumes that similar states can be grouped and result in the same behavior. This can also be understood as a single feature model, in which we assume that the observations can be described by a single feature, representing the clusters. Thus, the substates reduce to cluster indicators, i.e., they indicate which feature best represents the observed state. The number of latent states is then equal to the number of features. One possibility to infer this number is to utilize a Chinese Restaurant Process (CRP), giving rise to a Bayesian nonparametric model [197]. A similar model, where the state space is clustered according to the played actions, is proposed in [207].

5.5.4.3 Relation to the Proposed Model

Our model has the advantage, as opposed to the clustering-based approach and the unique coefficient model, that the features provide a means to understand the observed

behavior. In contrast to the unique coefficient model, we neither require binary coefficients nor a clustering step. Compared to the clustering-based approach, our model is able to significantly reduce the number of latent features and policies, as the features can be shared by different states. However, our model suffers from the assumption of a linear relation between features, substates, and policies. This problem is alleviated in the other approaches, as the features are decoupled from the policies. Note that our model becomes similar to the clustering-based model at the price of higher computational costs, if each substate is represented by only one feature.

5.6 Bayesian Nonparametric Model for Feature Learning

In this section, we provide a general framework for Bayesian nonparametric feature learning for decision-making based on the model proposed in Section 5.5.1. In order to learn the structure, we assume that we are given a set of observations consisting of state-action pairs, $\mathcal{D} = \{(\mathbf{z}_1, u_1), \dots, (\mathbf{z}_{N_z}, u_{N_z})\}$.

5.6.1 Observation Likelihood and Noise Variance

The observations, $\mathbf{z}_n, n = 1, \dots, N_z$, are assumed to be conditionally independent. As we assume Gaussian noise in Eq. (5.1), the state likelihood can be expressed as

$$p(\mathbf{Z} | \mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z^2) = \prod_{n=1}^{N_z} \mathcal{N}_{\mathbf{z}_n}((\mathbf{A} \odot \mathbf{W}) \mathbf{s}_n, \sigma_z^2 \mathbf{I}) . \quad (5.4)$$

with $\mathbf{Z} = [\mathbf{z}_1 \ \dots \ \mathbf{z}_{N_z}]$ and $\mathbf{S} = [\mathbf{s}_1 \ \dots \ \mathbf{s}_{N_z}]$. The variance of the noise, σ_z^2 , is assumed to be Inverse-Gamma distributed with hyperparameters $\alpha_\sigma, \beta_\sigma$. Further, we place hyperpriors on α_σ and β_σ , following Gamma distributions with hyperparameters $h_{\alpha_\sigma}^{(1)}, h_{\alpha_\sigma}^{(2)}, h_{\beta_\sigma}^{(1)}$, and $h_{\beta_\sigma}^{(2)}$.

5.6.2 Prior for the Feature Weights and Activations

As explained above, the prior probability of the feature weights, \mathbf{W} , depends on the problem at hand. We consider i.i.d. positive-valued feature weights, $w_{d,k}$ with $k =$

$1, \dots, K$ and $d = 1, \dots, D$, and assume an Exponential prior,

$$p(\mathbf{W} | \gamma_{\mathbf{w}}) = \prod_{k=1}^K \prod_{d=1}^D \text{Exp}_{w_{d,k}}(\gamma_{\mathbf{w}}).$$

The scaling factor, $\gamma_{\mathbf{w}}$, is assumed to be Inverse-Gamma distributed with hyperparameters α_{γ} and β_{γ} .

If the features are assumed to be real-valued, the prior can be modeled with a Gaussian distribution with straightforward modifications.

As explained, the feature activation matrix, \mathbf{A} , is modeled as IBP [15, 75]. This is detailed in Section 2.3.3.1.

5.6.3 Prior for the Substates

As formulated in Eq. (5.1), we assume that the observations are composed of a mixture of features weighted by the substates. Similar to a Factored Markov Decision Process (FMDDP)², we restrict the domain of the substate elements, $s_{k,n}$, with $n = 1, \dots, N_z$ and $k = 1, \dots, K$, to take values from a finite set, $\mathcal{S} = \{\check{s}_1, \dots, \check{s}_L\}$, where L denotes the number of elements, to simplify inference. For convenience, we assume equidistant elements in \mathcal{S} and set $\check{s}_1 = 0$ and $\check{s}_L = 1$. Note that the limited range does not restrict the model, as the features can be scaled to fit the observations.

Further, we assume that the substates are sparse, meaning that each observation consists of only a few features such that only few polices determine the observed action. In particular, we consider a sparsity-promoting mixture prior on the substates, similar to a *spike* and *slab* model [24, 25]. The components are given by a Categorical distribution, where $P(s = 0 | \theta_{s=0})$ is the sparsity component and $P(s \neq 0 | \theta_{s \neq 0})$ is the weight component. Note that all categories, except $\check{s}_1 = 0$, have equal probability. We place a Beta prior, $p(\boldsymbol{\theta}_s | \frac{\alpha_{s=0}}{2}, \frac{\alpha_{s \neq 0}}{2})$, with hyperparameters $\alpha_{s=0}$ and $\alpha_{s \neq 0}$, over the mixture

²A fundamental difference between our model and a FMDDP is that the substates in our model are, theoretically, not restricted to be finite as we do not need to enumerate over them.

weights $\boldsymbol{\theta}_s = \{\theta_{s=0}, \theta_{s \neq 0}\}$ with $\theta_{s \neq 0} = 1 - \theta_{s=0}$. Marginalizing over $\theta_{s=0}$ yields,

$$\begin{aligned} P(\mathbf{S}) &= \prod_{k=1}^K \int_0^1 \prod_{n=1}^{N_z} \left(P(s_{k,n} = 0 | \theta_{s=0}) \delta(s_{k,n}) \right. \\ &\quad \left. + P(s_{k,n} \neq 0 | \theta_{s=0}) (1 - \delta(s_{k,n})) \right) \\ &\quad \times p(\boldsymbol{\theta} | \alpha_{s=0}, \alpha_{s \neq 0}) d\theta_{s=0} \\ &\propto \prod_{k=1}^K \text{BetaBin}_{\mathbf{s}_k} (m_{s=0,k} + \alpha_{s=0}, m_{s \neq 0,k} + \alpha_{s \neq 0}) , \end{aligned}$$

where $\delta(s)$ returns one if $s = 0$ and zero otherwise, $m_{s=0,k}$ counts the zero elements in \mathbf{s}_k , and $m_{s \neq 0,k} = N_z - m_{s=0,k}$. The derivation of this expression can be found in Appendix B.3.

5.6.4 Mixture of Policies and Action Likelihood

Since we assume a finite set of actions, we consider a Categorical distribution for each mixture component,

$$P(u | \boldsymbol{\phi}_k) = \text{Cat}_u(\boldsymbol{\phi}_k) ,$$

such that the mixture model can be written as

$$P(u | \boldsymbol{\Phi}, \mathbf{s}) = \frac{1}{Z_u} \sum_{k=1}^K s_k P(u | \boldsymbol{\phi}_k), \quad (5.5)$$

with normalization constant $Z_u = \sum_{u \in \mathcal{U}} \sum_{k=1}^K s_k P(u | \boldsymbol{\phi}_k)$. The parameters of the policy of each mixture component follow Dirichlet distributions with identical hyperparameters, α_ϕ ,

$$p(\boldsymbol{\Phi}) = \prod_{k=1}^K \text{Dir}_{\boldsymbol{\phi}_k}(\alpha_\phi, \dots, \alpha_\phi) ,$$

where we assume independent policies. We consider the hyperparameter, α_ϕ , as a Gamma distributed variable with parameters $h_{\phi_A}^{(1)}$ and $h_{\phi_A}^{(2)}$.

As explained, we require a data set containing observed actions, u_n , $n = 1, \dots, N_z$. Assuming that the observed actions are independent, the likelihood is given as

$$P(\mathbf{u} | \boldsymbol{\Phi}, \mathbf{S}) = \prod_{n=1}^{N_z} \frac{1}{Z_{u_n}} \sum_{k=1}^K s_k P(u_n | \boldsymbol{\phi}_k).$$

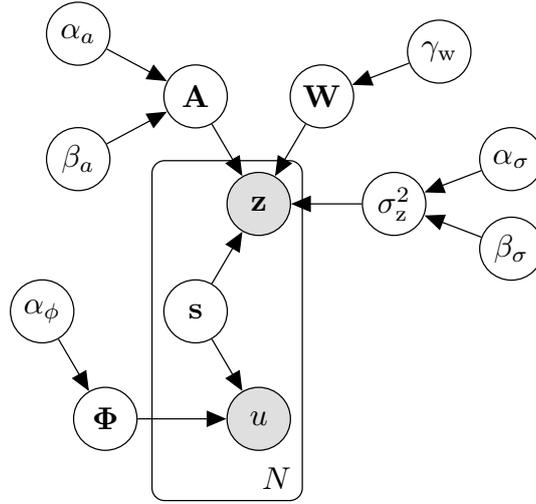


Figure 5.1. Graphical model of the feature-based decision-making model. Only the states, \mathbf{z}_n , and the actions, u_n , with $n = 1, \dots, N_z$, are observed. The other variables are latent and need to be inferred.

5.6.5 Joint Posterior Distribution

The joint posterior distribution can be factorized as

$$\begin{aligned}
 p(\mathbf{W}, \mathbf{A}, \mathbf{S}, \Phi, \sigma_z^2, \gamma_w, \alpha_a, \beta_a, \alpha_\sigma, \beta_\sigma, \alpha_\phi \mid \mathbf{Z}, \mathbf{u}) &\propto \\
 p(\mathbf{Z} \mid \mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z^2) P(\mathbf{u} \mid \mathbf{S}, \Phi) p(\sigma_z^2 \mid \alpha_\sigma, \beta_\sigma) & \\
 \times p(\mathbf{S}) p(\mathbf{W} \mid \gamma_w) P(\mathbf{A} \mid \alpha_a, \beta_a) p(\Phi \mid \alpha_\phi) & \\
 \times p(\gamma_w \mid \alpha_\gamma, \beta_\gamma) p(\alpha_\sigma) p(\beta_\sigma) p(\alpha_a) p(\beta_a) p(\alpha_\phi). &
 \end{aligned} \tag{5.6}$$

The conditional independences in this model are exploited in Section 5.7, where inference based on this model is explained. The structure of the posterior is illustrated as a graphical model in Fig. 5.1.

5.7 Inference

We consider the problem of learning the latent variables and the prediction of optimal actions for new observations as a Bayesian inference problem. Thus, we are interested in the joint posterior distribution in Eq. (5.6). Since the joint posterior is not directly tractable, we represent it by samples generated by means of Gibbs sampling. Therefore, in the first part of this section, we derive the conditional distributions of the variables. After explaining the sampling scheme in the second part, in the third part, we detail how to predict actions for new observations using this model.

For convenience, we use the bar symbol ($-$) in what follows to denote the set of conditional variables, i.e., all variables except the one that shall be sampled. The set of all latent variables is denoted by

$$\Omega = \{\mathbf{W}, \mathbf{A}, \mathbf{S}, \Phi, \sigma_z^2, \gamma_w, \alpha_a, \beta_a, \alpha_\sigma, \beta_\sigma, \alpha_\phi\} \in \Omega,$$

where Ω denotes the joint space of the latent variables.

5.7.1 Conditional Distributions

5.7.1.1 Sampling the Noise Variance

The hyperpriors of α_σ and β_σ are conjugate to the prior of σ_z^2 . Hence, the conditional $p(\sigma_z^2 | -)$ is also Inverse-Gamma distributed,

$$\begin{aligned} p(\sigma_z^2 | -) &\propto p(\mathbf{Z} | \mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z^2) p(\sigma_z^2 | \alpha_\sigma, \beta_\sigma) \\ &\propto \text{IGa}_{\sigma_z^2} \left(\alpha_\sigma + \frac{N_z D}{2}, \beta_\sigma + \frac{1}{2} \sum_{n=1}^{N_z} \sum_{d=1}^D \left(z_{d,n} - \sum_{k=1}^K a_{d,k} w_{d,k} s_{k,n} \right)^2 \right). \end{aligned}$$

The derivation can be found in Appendix B.2. We use a Metropolis-Hastings algorithm with a Gamma proposal distribution to generate samples of the hyperparameters, α_σ and β_σ .

5.7.1.2 Sampling the Substates

Sampling the substates, \mathbf{S} , is simple thanks to the assumption of a finite space of the elements. The conditional consists of the state and action likelihoods as wells as the (conditional) prior,

$$P(s_{k,n} | -) \propto p(\mathbf{z}_n | \mathbf{W}, \mathbf{A}, \mathbf{s}_n, \sigma_z^2) P(u_n | \mathbf{s}_n, \Phi) P(s_{k,n} | \mathbf{S} \setminus s_{k,n}), \quad (5.7)$$

for all $s_{k,n} \in \mathcal{S}$, where $\mathbf{S} \setminus s_{k,n}$ denotes the elements of \mathbf{S} without $s_{k,n}$. As the prior for the substates follows a Beta-Binomial distribution, the conditional is simply given as

$$P(s_{k,n} | \mathbf{S} \setminus s_{k,n}) \propto \begin{cases} m_{s=0} + \alpha_{s=0} & \text{for } s_{k,n} = 0 \\ m_{s \neq 0} + \alpha_{s \neq 0} & \text{for } s_{k,n} \neq 0 \end{cases},$$

where $m_{s=0}$ and $m_{s \neq 0}$ are defined in Section 5.6.3.

5.7.1.3 Sampling the Feature Weights

Since the likelihood is Gaussian and the weights are i.i.d. following an Exponential distribution, the conditional of the k th column of the weight matrix, \mathbf{w}_k , is a truncated Gaussian distribution,

$$\begin{aligned} p(\mathbf{w}_k | -) &\propto p(\mathbf{Z} | \mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z^2) \prod_{d=1}^D p(w_{d,k} | \gamma_w) \\ &\propto \mathcal{TN}_{\mathbf{w}_k}(\boldsymbol{\mu}_{\mathbf{w}_k}, \boldsymbol{\Sigma}_{\mathbf{w}_k}), \end{aligned}$$

with $\mathcal{TN}_{\mathbf{w}_k}(\boldsymbol{\mu}_{\mathbf{w}_k}, \boldsymbol{\Sigma}_{\mathbf{w}_k})$ denoting a truncated Gaussian, where the elements of \mathbf{w}_k are constraint to be positive valued. The mean, $\boldsymbol{\mu}_{\mathbf{w}_k}$, and the covariance, $\boldsymbol{\Sigma}_{\mathbf{w}_k}$, are given as

$$\boldsymbol{\Sigma}_{\mathbf{w}_k}^{-1} = \frac{1}{\sigma_z^2} \sum_{n=1}^{N_z} s_{k,n}^2 \text{diag}(\mathbf{a}_k) \quad (5.8)$$

$$\boldsymbol{\mu}_{\mathbf{w}_k} = \boldsymbol{\Sigma}_{\mathbf{w}_k}^{-1} \left(\frac{1}{\sigma_z^2} \sum_{n=1}^{N_z} s_{k,n} \left(\mathbf{z}_n - \sum_{\substack{k'=1 \\ k' \neq k}}^K s_{k',n} (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) \right) \odot \mathbf{a}_k - \mathbf{1}_D \frac{1}{\gamma_w} \right), \quad (5.9)$$

where the derivation can be found in Appendix B.1.2. Further, if we assume real-valued weights, a Gaussian prior can be utilized which is derived in Appendix B.1.3. We use the algorithm presented in [158] to sample from a truncated Gaussian distribution. Note that we neglect the dependency on the activations in Eq. (5.8) since the covariance would be infinite for $a_{k,d} = 0$. This does not affect the sampling scheme, as the corresponding weight will be ignored due to $a_{k,d} = 0$ anyway. Sampling the hyperparameter, γ_w , is fairly easy since the conditional of γ_w is an Inverse-Gamma distribution,

$$p(\gamma_w | -) \propto \text{IGa}_{\gamma_w} \left(\alpha_\gamma + \frac{KD}{2}, \beta_\gamma + \frac{1}{2} \sum_{k=1}^K \sum_{d=1}^D w_{d,k} \right).$$

5.7.1.4 Sampling the Feature Activations

Sampling from the IBP consists of two steps: For each row, (i) the active columns are updated and then (ii) new features are proposed. In the first step, an element of the activation matrix, $a_{k,d}$, is set active with probability

$$P(a_{d,k} | -) \propto p(\mathbf{z}_d | \mathbf{Sf}_d, \sigma_z^2) P(a_{k,d} | \mathbf{a}_{k \setminus d}), \quad (5.10)$$

with $\mathbf{a}_{k \setminus d}$ denoting the k th column of \mathbf{A} without the d th element and \mathbf{f}_d the d th row of \mathbf{F} . The derivation of the conditional, $P(a_{k,d} | \mathbf{a}_{k \setminus d})$, in Eq. (5.10) is detailed in Section 2.3.3.2 [15, 78]

In the second step, K'_+ new features are proposed in a Metropolis step [75, 160]. The proposal distribution, $q(\theta^+ | \theta)$, is independent of the previous sample, θ , as it consists of the priors for the substates, the feature weights, and the policies of the features,

$$q(\theta^+ | \theta) = q(\theta^+) = P(K'_+ | -)p(\mathbf{W} | \gamma_w)P(\mathbf{S})p(\Phi | \alpha_\phi) \quad (5.11)$$

with $\theta = \{\mathbf{W}, \mathbf{S}, \mathbf{A}, \Phi\}$ and $\theta^+ = \{\mathbf{W}^+, \mathbf{S}^+, \mathbf{A}^+, \Phi^+\}$, where \mathbf{W}^+ , \mathbf{S}^+ , \mathbf{A}^+ , and Φ^+ denote the proposed feature weights, activations, coefficients, and policies. The probability of adding K'_+ features is defined in Section 2.3.3.2 [15, 78].

As the proposal distribution is independent of the previous sample, the acceptance ratio, r , is equal to the likelihood ratio between the new and existing features [159],

$$r = \frac{P(\mathbf{Z} | \theta^+)}{P(\mathbf{Z} | \theta)}.$$

Since the IBP tends to mix slowly, we augment this ratio with probability P^+ of accepting a single new feature, resulting in a modified acceptance ratio which is derived in [75] and explained in Section 4.5.4. This increases the probability of proposing new features, leading to a faster convergence to the stationary distribution. The hyperparameters α_a and β_a are sampled as described in [75].

5.7.1.5 Sampling the Policies

Sampling from the conditionals for the policies directly is difficult and would be computationally expensive due to the mixture model (Eq. (5.5)). An efficient approach is to introduce auxiliary variables, t_n , $n = 1, \dots, N_z$, for each observation, indicating from which policy the observed action, u_n , has been generated [208]. Given the indicators, the mixture components ϕ_k , $k = 1 \dots, K$, in Eq. (5.5) become conditionally independent of the mixture weights, \mathbf{s}_n , $n = 1, \dots, N_z$, which makes sampling the components straightforward. The sampling algorithm thus consists of two steps.

First, the indicators are sampled according to

$$P(t_n = k | u_n, \mathbf{S}, \phi) \propto s_k P(u_n | \phi_k). \quad (5.12)$$

In order to approximate the conditional for the policies, we draw N_t indicator samples from Eq. (5.12). Drawing the samples is easy, since the indicators follow Categorical distributions with $t_n \in \{1, \dots, K\}$.

Second, given the indicators, the parameters of the k th feature policy, ϕ_k , is sampled from a Dirichlet-Multinomial distribution,

$$p(\phi_k | \mathbf{u}, \mathbf{t}) = \text{DirMult}_{\phi_k}(m_{\phi_k,1} + \alpha_\phi, \dots, m_{\phi_k, N_u} + \alpha_\phi)$$

where $m_{\phi_k, i}$, $i = 1, \dots, N_u$, counts the co-occurrences between the policy indicators, \mathbf{t} , and the actions, $u_n \in \mathcal{O}_u$.

The hyperparameter, α_ϕ , can be sampled in a Metropolis step, using a Gamma proposal distribution.

5.7.2 Sampling Algorithm

The Gibbs sampler is initialized with only one feature and the variables are sampled from their prior distributions. After several iterations of the Gibbs sampler, samples from the target distribution are generated.

Note that there is the chance to generate new features in each iteration of the Gibbs sampler. Especially in scenarios with strong noise, different rows of the feature matrix may converge to similar realizations, increasing the number of features unnecessarily. We propose to merge features reducing the number of features, if they show a similarity larger than a prefixed threshold, T_{corr} , where we keep the activations of both features and average the policies. The similarity is measured by means of the estimated correlation between the feature samples.

A MAP estimator can be utilized, if we are interested in an estimate of the latent variables, Ω_{MAP} , containing, i.a., estimates of the features, $\hat{\mathbf{F}}_{\text{MAP}}$ [209], and the policies, $\hat{\Phi}_{\text{MAP}}$. The MAP estimator can be approximated by choosing the sample with the highest posterior probability. The posterior probability is calculated according to Eq. (5.6). For the prediction of actions for new observations, we can also use a MMSE estimator which provides better generalization capabilities. This is detailed in the next section.

5.7.3 Prediction of Actions

The proposed model can be used to learn the structure of the observed states as well as for the prediction of actions, given new observed states. For the prediction of an optimal action, u^* , given a new observation, \mathbf{z}^* , we can evaluate the posterior predictive distribution, giving rise to a MMSE estimator,

$$P(u^* | \mathbf{z}^*, \mathcal{D}) = \int_{\mathcal{S}} \int_{\Omega} P(u^*, \mathbf{s}^* | \mathbf{z}^*, \Omega) p(\Omega | \mathbf{z}^*, \mathcal{D}) d\Omega d\mathbf{s}^*, \quad (5.13)$$

where we exploit that u^* is independent of the data set containing the observations, \mathcal{D} , given Ω . Eq. (5.13) shows that Ω depends on the new observations, \mathbf{z}^* . Thus, all variables would need to be inferred for each prediction, which would be computationally expensive. To remedy this issue, we assume that the observed data in \mathcal{D} sufficiently represents the conditional distribution for Ω , such that we can ignore the dependency and simply infer Ω based on \mathcal{D} , leaving only u^* and \mathbf{s}^* to be inferred during prediction. The marginalization over Ω is approximated by Monte Carlo integration. For this, we need to draw samples of \mathbf{s}^* given the samples of Ω . The samples can be generated by drawing from the conditional in Eq. (5.7). Alternatively, we obtain a MAP estimate of u^* by maximizing the posterior predictive distribution with respect to \mathbf{s}^* given Ω_{MAP} .

Since, as justified in Section 5.5.3, we assume a deterministic policy, the optimal action, u_{opt}^* , is the action that maximizes $P(u^* | \mathbf{z}^*, \mathcal{D})$. Thus, the posterior predictive distribution expresses our confidence about the actions and can be considered as the policy of the agent.

Note that, especially when we are interested in the prediction of actions, a modification on the model can help to increase the prediction accuracy dramatically. Especially with high-dimensional observations, the observation likelihood determines the posterior, where the effect of the action likelihood nearly vanishes, leading in the worst case to a neglect of the actions. A remedy consists in considering an action variable for each entry of the observation. However, we assume that these actions variables are identical. This increases the influence of the action log-likelihood by factor D , increasing the weight on the posterior significantly. The necessary modification in the inference algorithm are simple: First, for the conditional of Eq. (5.7), the action log-likelihood is multiplied by D . Second, the same modification is applied to the joint posterior distribution in Eq. (5.6). Sampling the policies is not affected as the actions for each substate are identical. We apply this modification on the model for the real data experiments in Section 5.9, as the observations in the experiments are high-dimensional.

Table 5.1. Parameter settings for the algorithm used in the simulation experiments

Parameter	Value	Meaning
$h_{\alpha_\sigma}^{(1)}$	1000	} hyperparameters for σ_z $h_{\alpha_\sigma}^{(1)}$ is chosen assuming a high SNR scenario
$h_{\alpha_\sigma}^{(2)}$	1	
$h_{\beta_\sigma}^{(1)}$	1	
$h_{\beta_\sigma}^{(2)}$	1	
$h_{\alpha_A}^{(1)}, h_{\alpha_A}^{(2)}$	1	hyperparameters for α_a
$h_{\beta_A}^{(1)}$	1	hyperparameter for β_a
$h_{\beta_A}^{(2)}$	10	hyperparameter for β_a
$\alpha_\gamma, \beta_\gamma$	1	hyperparameters for \mathbf{W}
$h_{\phi_A}^{(1)}, h_{\phi_A}^{(2)}$	1	hyperparameters for Φ
α_s, β_s	1	hyperparameters for \mathbf{S}
P^+	0.01	probability of accepting $K'_+ = 1$ features
T_{corr}	0.9	threshold for merging similar features
N_{iter}	10000	number of iterations of the Gibbs Sampler
L	100	size of \mathcal{S}
N_t	1000	number of samples of the policy indicators

5.8 Experimental Results

In order to demonstrate the performance of the proposed method, we consider simulations as well as real data experiments. In this section, we evaluate the performance of the inference algorithm by simulating observations with different SNRs and different latent numbers of features. For this, we simulate the ground truth values by drawing samples from distributions which are similar to the prior distributions of the variables. This is detailed in the following. The true hyperparameters of the features weights are set to $\check{h}_{\alpha_\gamma} = \check{h}_{\beta_\gamma} = 100$. Thus, the true hyperparameter for the weights $\check{\gamma}_w$ as well as the true weights $\check{\mathbf{W}}$ are sampled from their prior distributions. An element of the true feature activation matrix, $\check{\mathbf{A}}$ is activated with probability $P(\check{a}_{k,d} = 1) = 0.5$. The true substates, $\check{\mathbf{s}}_n$, are simulated by drawing samples from a Dirichlet distribution with parameters $\frac{1}{K}\mathbf{1}_K$, resulting in a peaked distribution³. The true noise variance, $\check{\sigma}_z^2$, is determined by the chosen SNR, where the signal power is estimated from $\check{\mathbf{X}} = \check{\mathbf{F}}\check{\mathbf{S}}$. The parameters of the ground truth policies, $\check{\Phi}$, are chosen such that one action has high probability mass, reflecting deterministic policies, as justified in Section 5.5.3. In all simulations, we consider $N_z = 100$ observations of dimension $D = 30$ with $N_u = 4$ different actions. In order to evaluate the prediction performance of the model, we split the data set into a training and a test data set, leaving 80 observations for training

³Note that the assumption of a Categorical distribution for the substates introduces a bias in the simulation results. However, increasing the number of categories alleviates this problem.

and 20 for testing. We sweep the SNR from 10 dB to 30 dB with a step size of 5 dB and vary the number of features K within the set of $\{5, 7, 9, 12, 15, 18\}$. We set the hyperparameters of the algorithm according to Tab. 5.1.

We organize the evaluation in three parts. First, we investigate the performance of the estimation of the features, the feature coefficients, and the reconstruction of the states. Second, we compare the estimated policies to the true policies and evaluate the prediction in terms of the accuracy, A_u , which is the average rate of correctly predicted actions of the test data set. These estimates are obtained utilizing a MAP estimator. As for the prediction of actions we can also use the MMSE estimator, as detailed in Section 5.7.3, we also discuss the results obtained by this estimator. Third, we provide results for the inferred number of features and compare it with the true value.

5.8.1 Estimation of the Features

The quality of the MAP estimates is evaluated in terms of the Mean Squared Error (MSE) for the elements of the features, \mathbf{F}_{MAP} , the substates, \mathbf{S}_{MAP} , and the reconstructions, $\mathbf{X}_{\text{MAP}} = \mathbf{F}_{\text{MAP}}\mathbf{S}_{\text{MAP}}$. Further, we compute the RMSEs over 20 Monte Carlo runs, yielding RMSE measures for each estimated variable, $\epsilon_{\mathbf{F}}$, $\epsilon_{\mathbf{S}}$, and $\epsilon_{\mathbf{X}}$. The results are shown in Fig. 5.2.

We obtain good results with low errors especially for a small latent number of features, almost independent of the SNR. With increasing K , we observe a strong growth of the error. The error becomes even more significant in case of strong noise. It is remarkable that the error of the estimated feature values as well as the feature coefficients behave similarly. Due to the linear relation between the features and the substates, the errors of the reconstructions also grows with the number of features.

5.8.2 Prediction of Actions

We evaluate the correctness of the estimated policies in terms of the RMSE of the Mean Absolute Deviation (MAD) over 20 Monte Carlo runs. As shown in Fig. 5.3, the error slightly grows with an increasing number of features. Again, the SNR has, compared to the number of features, only little effect on the accuracy of the policies.

The accuracy of the action prediction using the MAP and MMSE estimators are depicted in Fig. 5.4. In case of few features ($K < 15$) and low noise (SNR > 20 dB), we

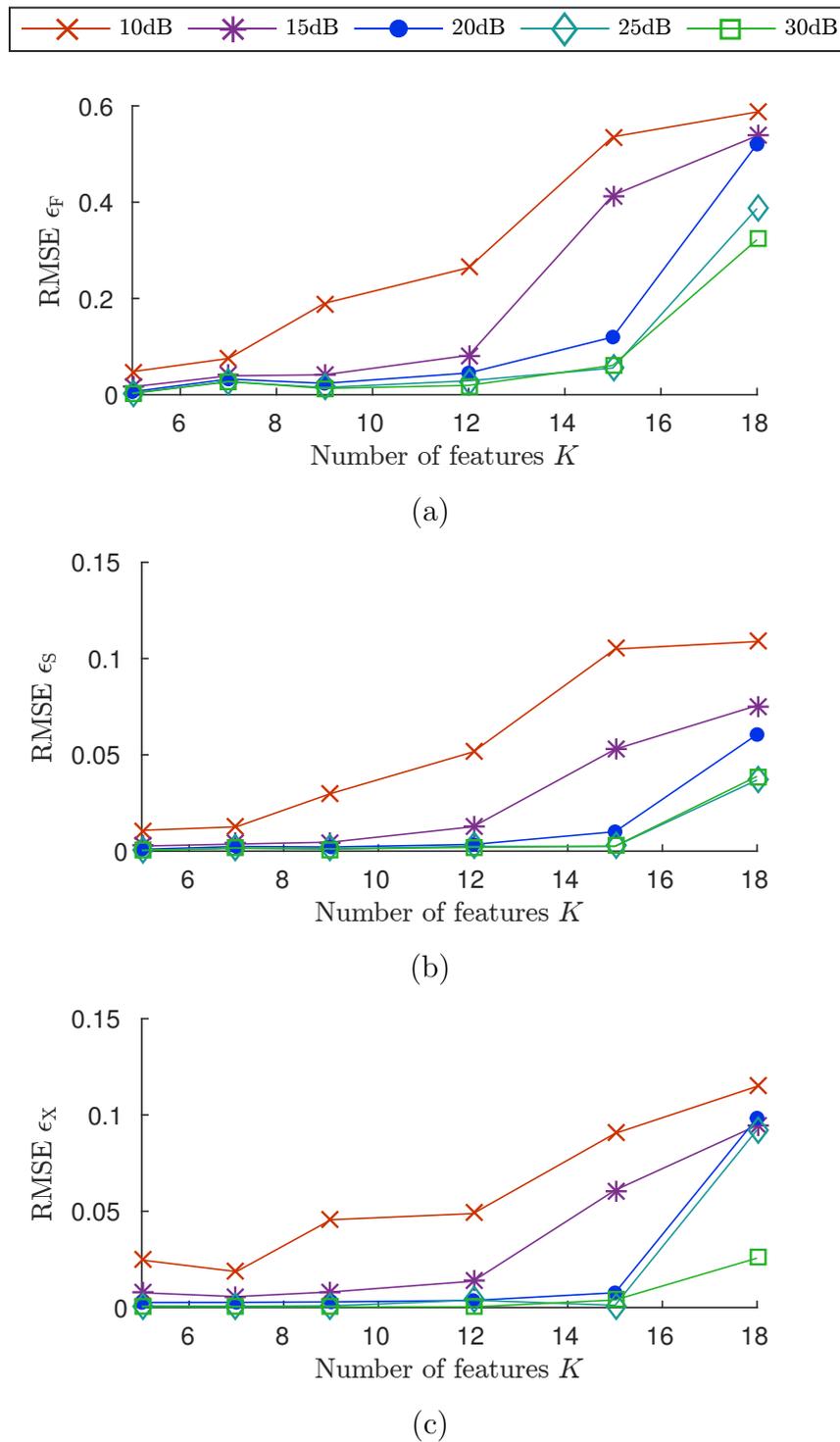


Figure 5.2. Results of the feature reconstruction. Shown are the RMSEs for (a) the features, (b) the substates, and (c) the reconstructed observations. All variables are reliably inferred in case of moderate to high SNRs (SNR > 15 dB). If the SNR drops below 15 dB, a reliable reconstruction of more than seven features cannot be guaranteed using the simulation settings.

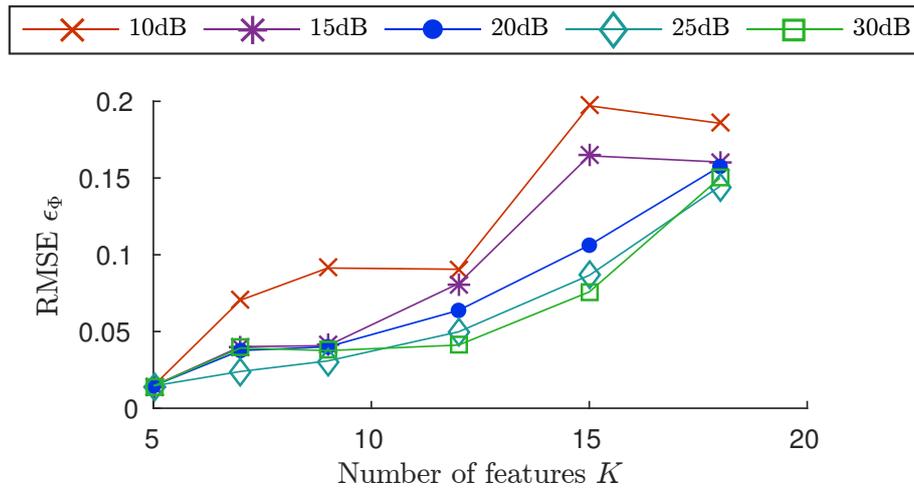


Figure 5.3. Results of the policies and features estimation. The error grows significantly with the number of features. For low numbers of features, the policies can be accurately recovered, almost independent of the SNR.

obtain highly accurate predictions with over 90 % accuracy. Only for strong noise and many latent features the accuracy drops slightly below 70 %. This observation can be explained by the relation between the action and substates, which are challenging to infer in case of many features.

Since we assume in the simulation experiments that there is a fixed set of parameters that explains the observations, the MMSE estimator yields only minor improvements over the MAP estimator concerning the accuracy of predicted actions, as indicated by the dotted lines in Fig. 5.4.

5.8.3 Estimation of the Number of Features

Assuming that the observations have been generated by a fixed number of features, we evaluate how accurately the algorithm is able to infer this number. The results for the simulations are depicted in Fig. 5.5, showing the RMSE of the MAD over the 20 Monte Carlo runs. As can be observed, the MAP estimator is able to infer the correct number of features reliably, especially in case of few features. On the one hand, if the noise in the observations increases and the observations are based on many latent features, the error grows significantly. On the other hand, if the SNR is reasonably high, i.e., $\text{SNR} > 15$ dB, the results deviate on average by only four from the true number in case of 18 latent features. The error in estimating the number of features is probably due to the fact that some simulation examples can be explained by less features than used for their generation.

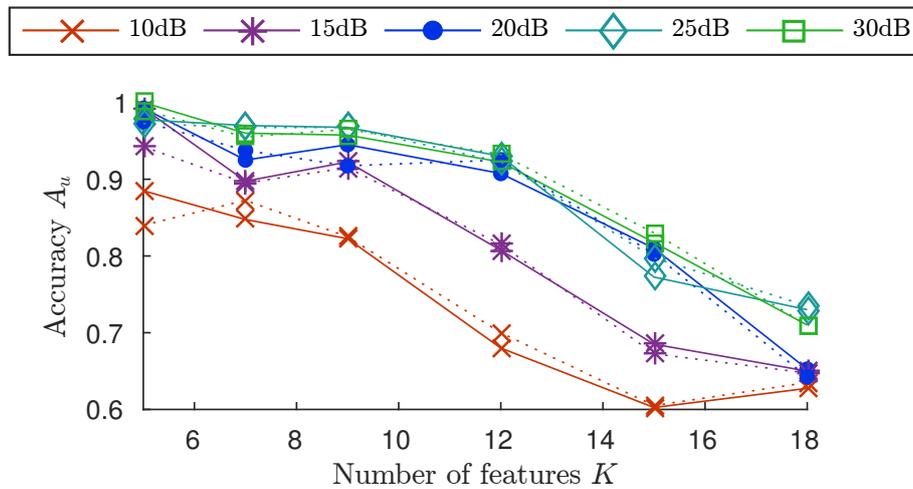


Figure 5.4. Accuracy of the predicted actions for a hold-out test data set using the MAP and MMSE estimators (solid and dotted lines). High accuracies are achieved in case of only few features. The prediction accuracy clearly suffers from strong noise, especially for many features. As the underlying structure can be explained by fixed parameters, the MMSE yields only little improvements over the MAP estimates.

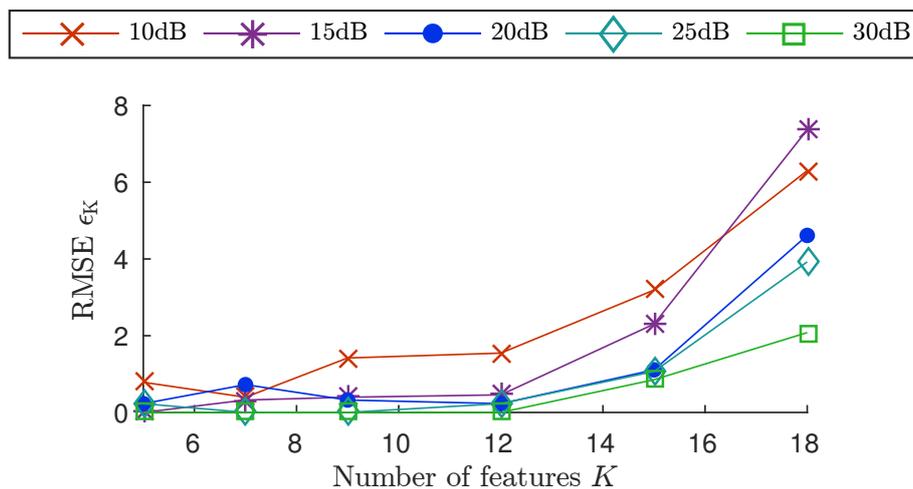


Figure 5.5. Results of the estimation of the number of features. The number is accurately estimated in most cases. Only in case of strong noise (SNR < 10 dB) and many latent features ($K > 15$), the error significantly increases.

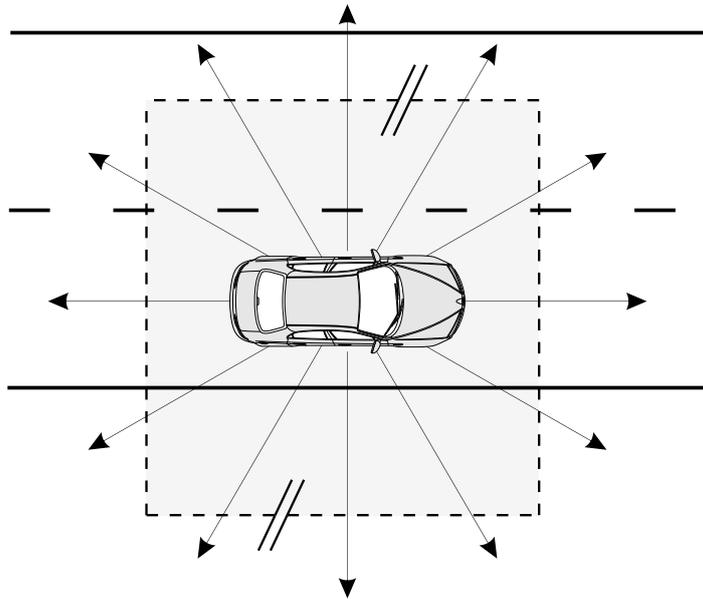


Figure 5.6. Illustration of the setup for the real data experiments. During pre-processing, we map the LIDAR measurement (indicated by the arrows) to an occupancy grid which is illustrated by the gray area surrounding the vehicle.

5.9 Real Data Experiments

We consider the problem of analyzing a driver’s behavior, which is an important task for user-adaptive driver assistance systems [210,211], in order to demonstrate the performance of the proposed model in a real-world scenario. For this, we observe the surrounding of the vehicle and the actions taken by the driver, aiming at learning what caused the driver to make the observed decisions. Using the proposed model, we can also predict which action the driver is likely to take given a certain situation. Thus, we also investigate the predictive performance of our approach by randomly creating training and test data sets. For this, we consider real data provided by the KITTI Vision Benchmark Suite [212] containing several challenges in urban driving. We use the data for the tracking challenge as it contains time-sequential LIDAR measurements of different situations in public road traffic. A schematic plot of the setup is illustrated in Fig. 5.6. We consider Scene 11 and 20 of the benchmark suite, which are detailed below in Section 5.9.1 and Section 5.9.2. Before running the proposed inference algorithm, we pre-process the data as follows. First, we apply a thresholding on the height values of the measurements such that we keep only samples above ground level, which is roughly 1.5 m below the LIDAR. Afterwards, we discretize the measurements to obtain positive-valued occupancy grids, where the values of the grid elements refer to the maximum measured height.

As observation in the n th time frame, we consider the joint occupancy grid of the current and the previous frame. This is required to implicitly include velocity information, which enables to decide, e.g., if the host vehicle is faster or slower than the vehicle in front. Thus, the minimum speed, v_{\min} , between the host vehicle, H , and an obstacle, O , that can be resolved depends on the resolution, R_G , of the grid as follows,

$$R_G \leq |d_{HO,n} - d_{HO,n-1}| = (t_{F,n} - t_{F,n-1}) |v_H - v_O| = \frac{1}{f_s} v_{\min},$$

where $d_{HO,n}$ denotes the distance between the host vehicle and the obstacle in the n th time frame. The time stamp is denoted by $t_{F,n}$ and f_s is the frequency at which the measurements are sampled (in the KITTI data set, $f_s = 10$ Hz).

We consider the environment of the vehicle in the range from -3 m to 3 m in the lateral direction. This covers the lane of the vehicle plus parts of (if existent) neighboring lanes. In Scene 11, the longitudinal range is limited between -10 m and 30 m and for Scene 20 between -40 m and 40 m. The range in longitudinal direction gives a time window of more than 2 s to react to the observed situation when driving at 50 km/h, which is the speed limit in German cities.

For both scenes, we choose a grid size of 21×65 pixels, resulting in a spatial resolution in lateral direction of $R_{G,\text{lat}} \approx 0.3$ m and in longitudinal direction of $R_{G,\text{long}} \approx 0.6$ m for Scene 11 (and $R_{G,\text{long}} \approx 1.2$ m for Scene 20). Thus, the minimum velocities that can be detected are $v_{\text{lat},\min} > 3$ m/s in the lateral direction and $v_{\text{long},\min} > 6$ m/s in the longitudinal direction (and $v_{\text{long},\min} > 12$ m/s for Scene 20).

We labeled the scenes to obtain the observed actions by means of the measured acceleration of the host vehicle, obtained from the onboard inertial measurement unit. The ground truth for Scene 11 is depicted in Fig. 5.7 and for Scene 20 in Fig. 5.11. The set of actions, \mathcal{O}_u , we consider in this experiment is comprised of *acceleration*, *deceleration*, *lane change (right)*, and moving at *constant speed*. However, due to the short duration of the sequences, usually fewer actions are observed in each scene.

We use the same settings of the hyperparameters as in Tab. 5.1. Only one of the two hyperparameter for the IBP, $h_{\alpha_A}^{(2)}$, is modified to reduce the number of expected features, $h_{\alpha_A}^{(2)} = 10$. Further, we perform 500 iterations of the Gibbs sampler. In both data sets, after a mere 100 iterations, the sampler converges to the stationary distribution and produces samples from the target distribution.

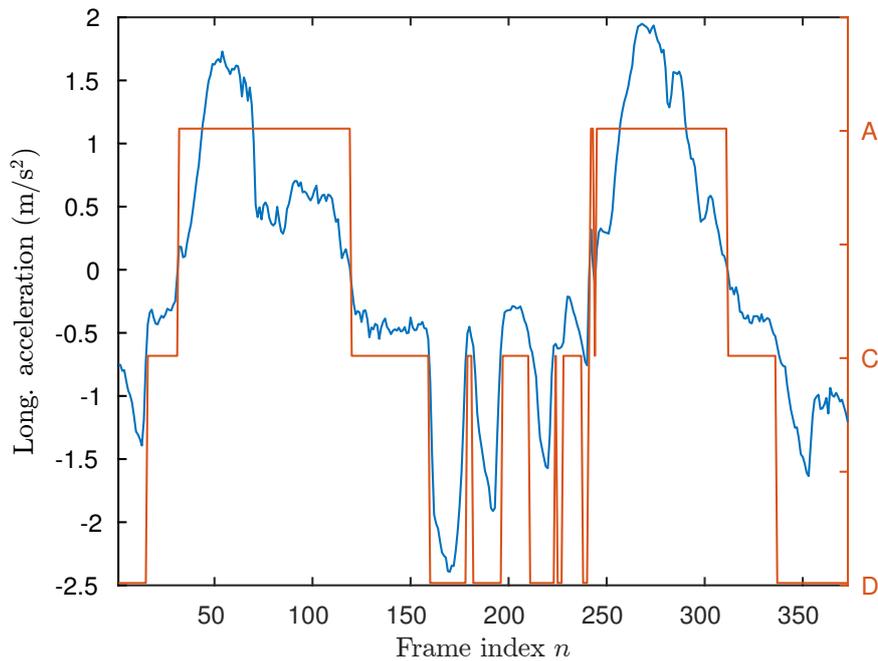


Figure 5.7. Ground truth labels of Scene 11. The actions *acceleration* (A), *moving at constant speed* (C), and *deceleration* (D) are chosen according to the acceleration of the vehicle.

5.9.1 Scene 11 - Traffic Jam

Scene 11 of the KITTI data set shows an urban scenario, in which the driver follows another vehicle. The vehicle in front cannot be overtaken due to a single-lane road. As shown in Fig. 5.7, the driver first accelerates. After a few seconds, the driver has to decelerate until the car stops due to a halt of the preceding car. When the vehicle in front starts moving again, the host vehicle accelerates.

Applying the proposed algorithm to the observations results in 31 features using the MAP estimator.

For the analysis of the features, we only visualize four of the most relevant features, which are selected according to the confidence in the corresponding feature policy.

Considering the current and the previous frames as inputs, we obtain feature estimates of consecutive frames. Observing the differences between the frames can be difficult as the features are likely to be highly similar due to the low temporal difference. Therefore, we show the feature of the current time frame and the difference of the features. A difference plot can be interpreted as the temporal gradient of two consecutive frames.

Thus, from the depicted patterns we can draw conclusions about the relative speed of the obstacles. For instance, if the difference plot shows negative values left of positive values (red-blue pattern), the obstacle is faster than the host vehicle. In contrast, a blue-red pattern indicates a slower vehicle. The width of the bars reflect the magnitude of the relative velocities.

The features are illustrated in Fig. 5.8 to Fig. 5.10. In the figures, the x -axis corresponds to the longitudinal and the y -axis to the lateral direction. Shown is the top-view on the host vehicle, which is indicated by the red cross. The intensity reflects the measured heights at each pixel. As explained, the difference features are obtained by subtracting the features of the previous from the current time frame.

The features explaining acceleration maneuvers are shown in Fig. 5.8. Features A1–A3 clearly show that the driver accelerates as long as the preceding car is significantly faster. A4, however, indicates acceleration though the preceding car decelerates. Since the car seems to be sufficiently far away, the driver has not yet decided to reduce the speed.

Fig. 5.9 depicts features indicating deceleration. Features D1–D3 explain the deceleration of the driver with a slower car in front. In contrast, D4 shows that the preceding car accelerates. However, due to the low distance between both cars, the driver decides to decelerate.

The features indicating moving at constant speed are shown in Fig. 5.10. Features C1–C3 show a vehicle in front of the driver’s car. The difference images reveal only little differences between the velocities of both vehicles, such that the driver maintains the current velocity. C4 represents an empty road, where, again, the driver does not need to adapt the speed of his car.

In order to evaluate the prediction performance, as in the simulations, 20% of the observations are used for testing while the rest is used for inferring the structure. Using the MAP estimator results in an accuracy of 74.32 %. The confusion matrix in Tab. 5.2 shows that, most notably, in some cases moving at constant speed and acceleration are confused. Further, deceleration is in few cases misclassified as acceleration.

Quantifying the results of the state reconstruction is difficult, as a ground truth is not available. To provide at least an intuition about the quality of the reconstructions, we compute the MSE between the reconstructed states, based on the estimated substates and features, and the observations. This results in a MSE of approx. 0.0329. As the values of the observations are in a similar range as in the simulations, this result

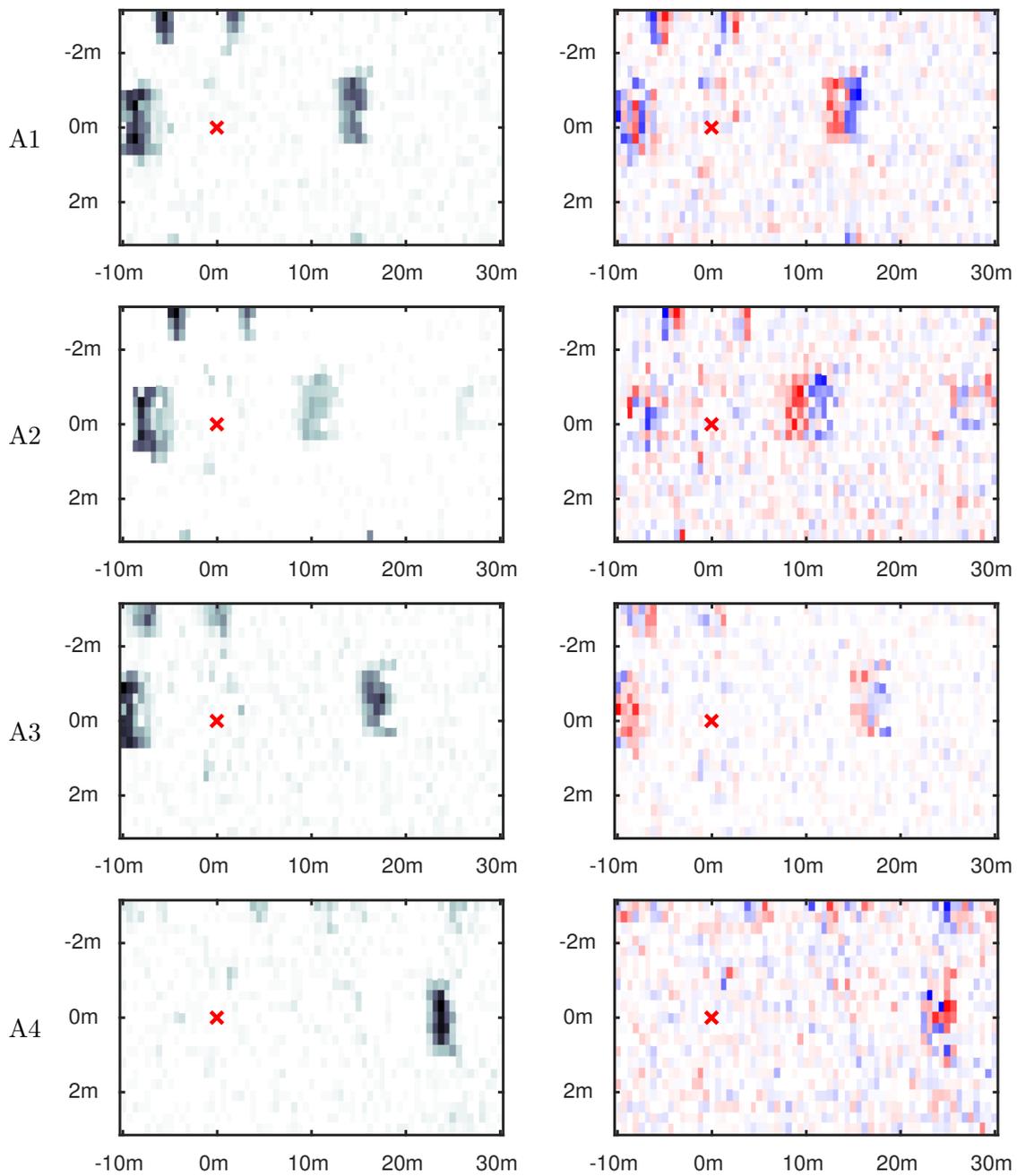


Figure 5.8. Features estimated from KITTI Scene 11 for action A , *acceleration*, (left) and the difference image (right), where red indicates negative and blue positive values. Features A1–A3 show an obstacle in front of the host vehicle, which is moving significantly faster as indicated by the difference plots. A4 shows an obstacle that slows down. As it is sufficiently far away, the driver still accelerates.

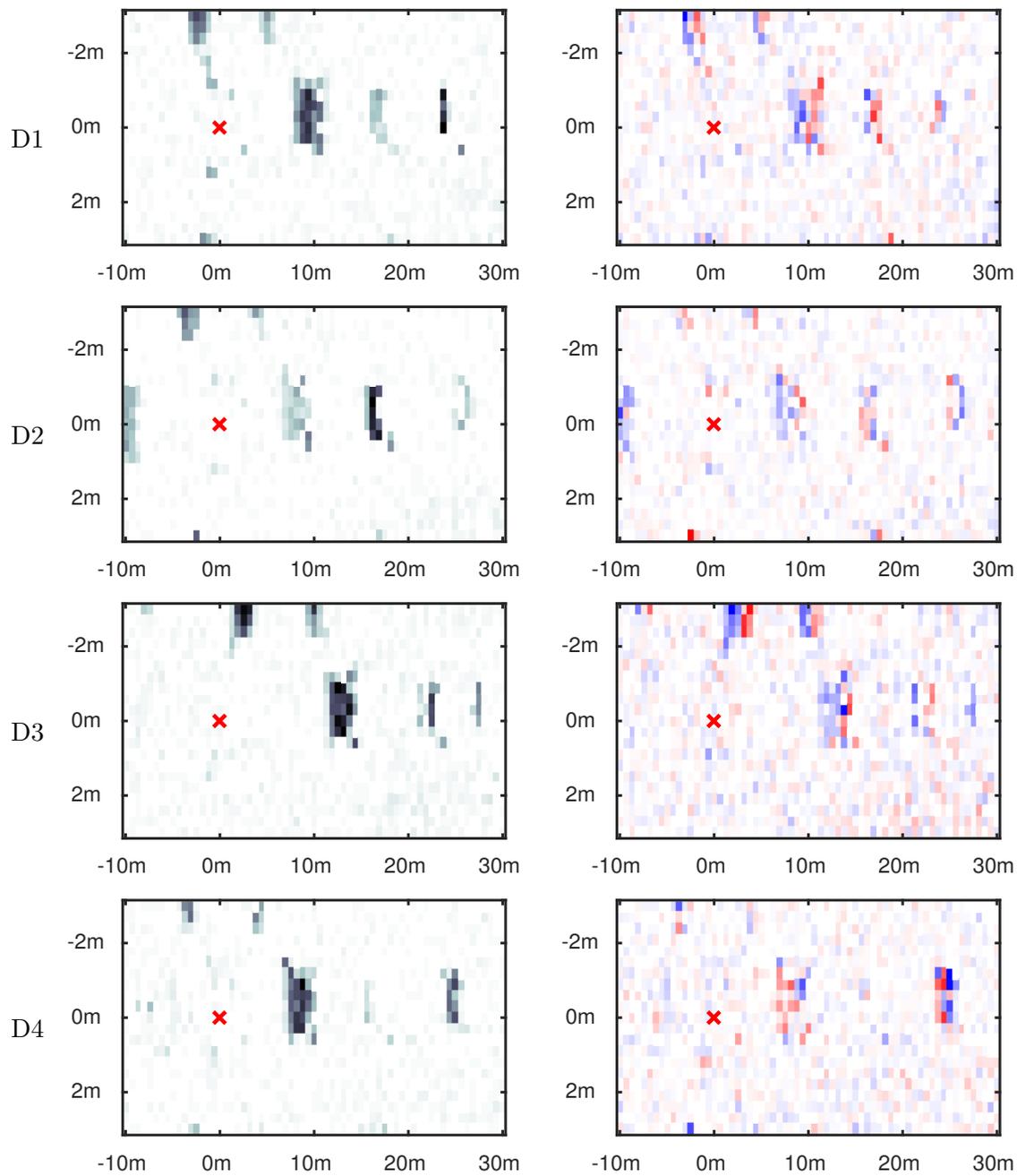


Figure 5.9. Features estimated from KITTI Scene 11 for action D, *deceleration*, (left) and the difference image (right), where red indicates negative and blue positive values. Features D1–D3 show vehicles in front of the host car, which are moving at a slower speed. D4 shows that the preceding car accelerates. As the host vehicle is close to the obstacle, the driver decelerates.

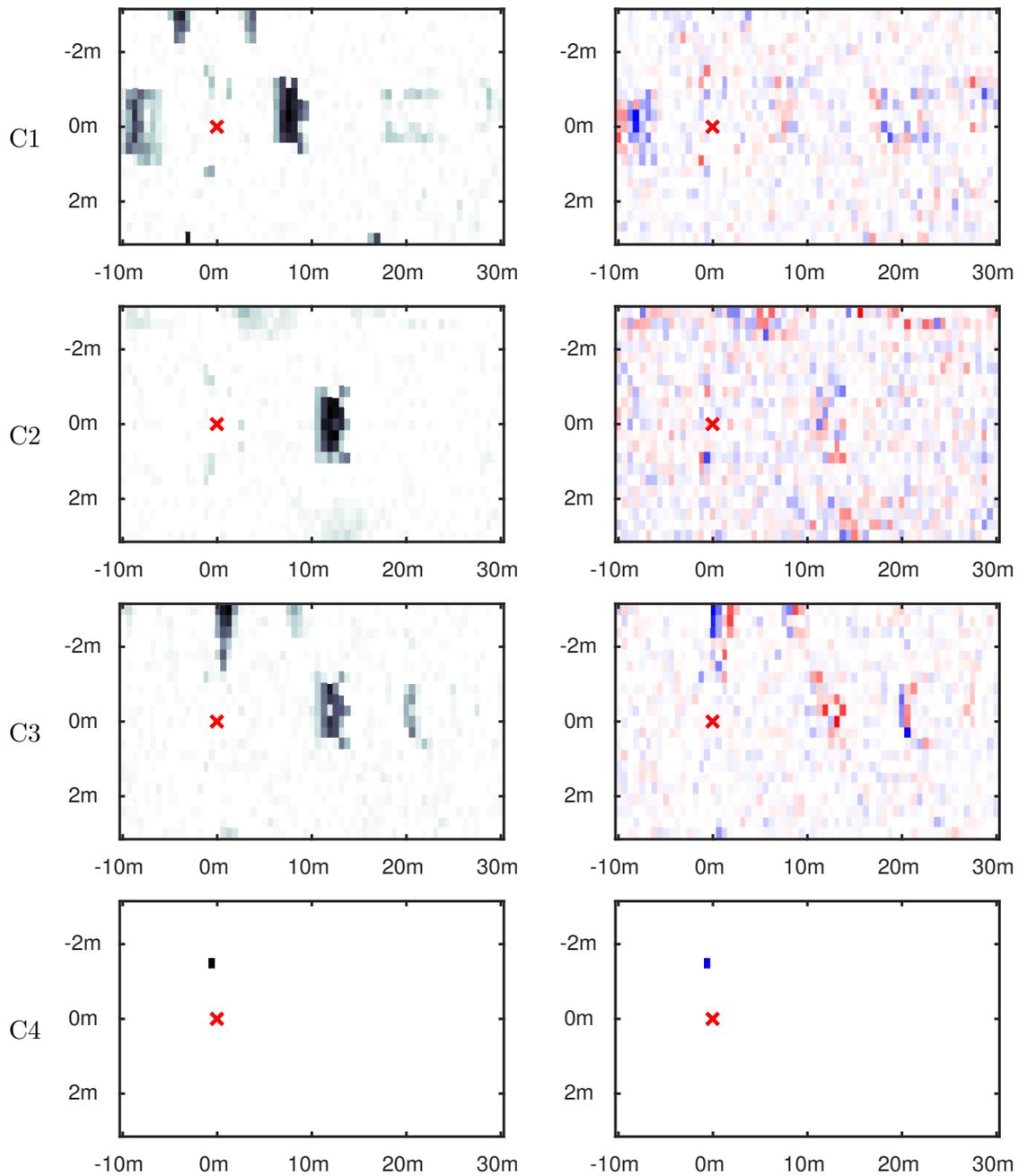


Figure 5.10. Features estimated from KITTI Scene 11 for action C , *moving at constant speed*, (left) and the difference image (right), where red indicates negative and blue positive values. Features $C1$ – $C3$ show preceding vehicles. As the difference plots do not indicate significant speed differences, the preceding and the host car travel at a similar speed, such that the driver does not need to adapt the speed. $C4$ shows an empty road without any obstacles.

Table 5.2. Confusion matrix for the hold-out data set of Scene 11. The overall accuracy is 74.32%, where 31 features have been inferred.

Ground truth	Prediction		
	Const. speed	Deceleration	Acceleration
Const. speed	12	2	4
Deceleration	1	15	6
Acceleration	5	1	28

Table 5.3. Confusion matrix for the hold-out data set of Scene 20. The overall accuracy is 73.33%, where 17 features have been inferred.

Ground truth	Prediction		
	Acceleration	Lane change (right)	Const. speed
Acceleration	3	3	7
Lane change (right)	0	29	0
Const. speed	0	6	12

indicates low errors, resembling the results of the simulations. Still, the number has to be taken with care, as we compare the denoised reconstruction with noisy observation.

5.9.2 Scene 20 - Lane Change

Scene 20 shows a lane change maneuver on a two-lane road. As can be observed from the acceleration signals depicted in Fig. 5.11, the driver accelerates three times, depending on the current traffic situation. The lane change takes place from time frame 158 to 220.

Applying the proposed algorithm reveals 17 features. Using these features to predict the actions of the test data set yields an accuracy of 73.33%. The confusion matrix in Tab. 5.3 shows that lane changes are reliably predicted. Acceleration maneuvers are in some cases misinterpreted as moving at constant speed or as a lane change. A third of the moving at constant speed observations are misclassified as lane changes. Comparing the reconstructed states with the noisy observation yields a MSE of 0.0027.

For illustration of the inferred features, Fig. 5.12 shows three out of the 17 features, each indicating a different action. The first feature (a) shows a lane change (right). As the driver is already performing the maneuver, the scene is rotated. The dark areas in the lower left corner represent vehicles behind the host car. The second feature (b) contains a vehicle behind the driver's car and another vehicle on the left lane. Due to

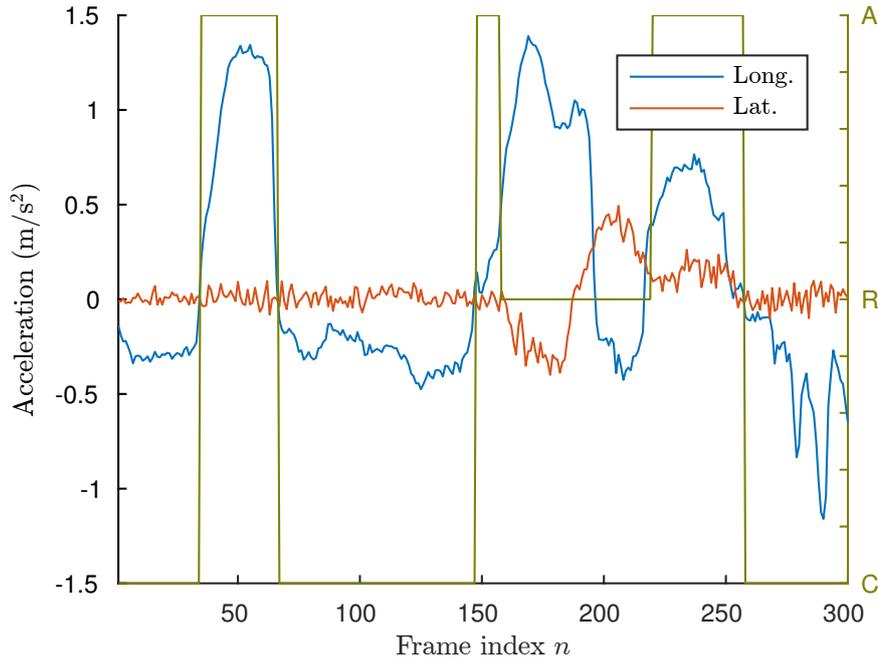


Figure 5.11. Ground truth labels of Scene 20. The actions *acceleration* (A), *lane change (right)* (R), and *moving at constant speed* (C) are chosen according to the acceleration of the vehicle.

the high traffic density, the driver does not accelerate. In contrast, the third feature (c) shows a completely empty road, motivating the driver to accelerate the vehicle.

5.10 Discussion

As shown in the simulation experiments, the algorithm based on the proposed model is able to reliably infer the number of features, the features and the policies. In case of strong noise, the algorithm finds several, almost equally probable explanations for the observations, resulting in variations of the MAP estimate. Especially when the number of features is high compared to the dimension of the observations, inferring the correct number of features can be challenging.

The real data experiments show that the model is able to provide deeper insights into the observations which may yield new conclusions about the observed behavior. As explained, in high-dimensional observations, the observation likelihood is likely to dominate the posterior leading to only little influence of the action likelihood and, hence,

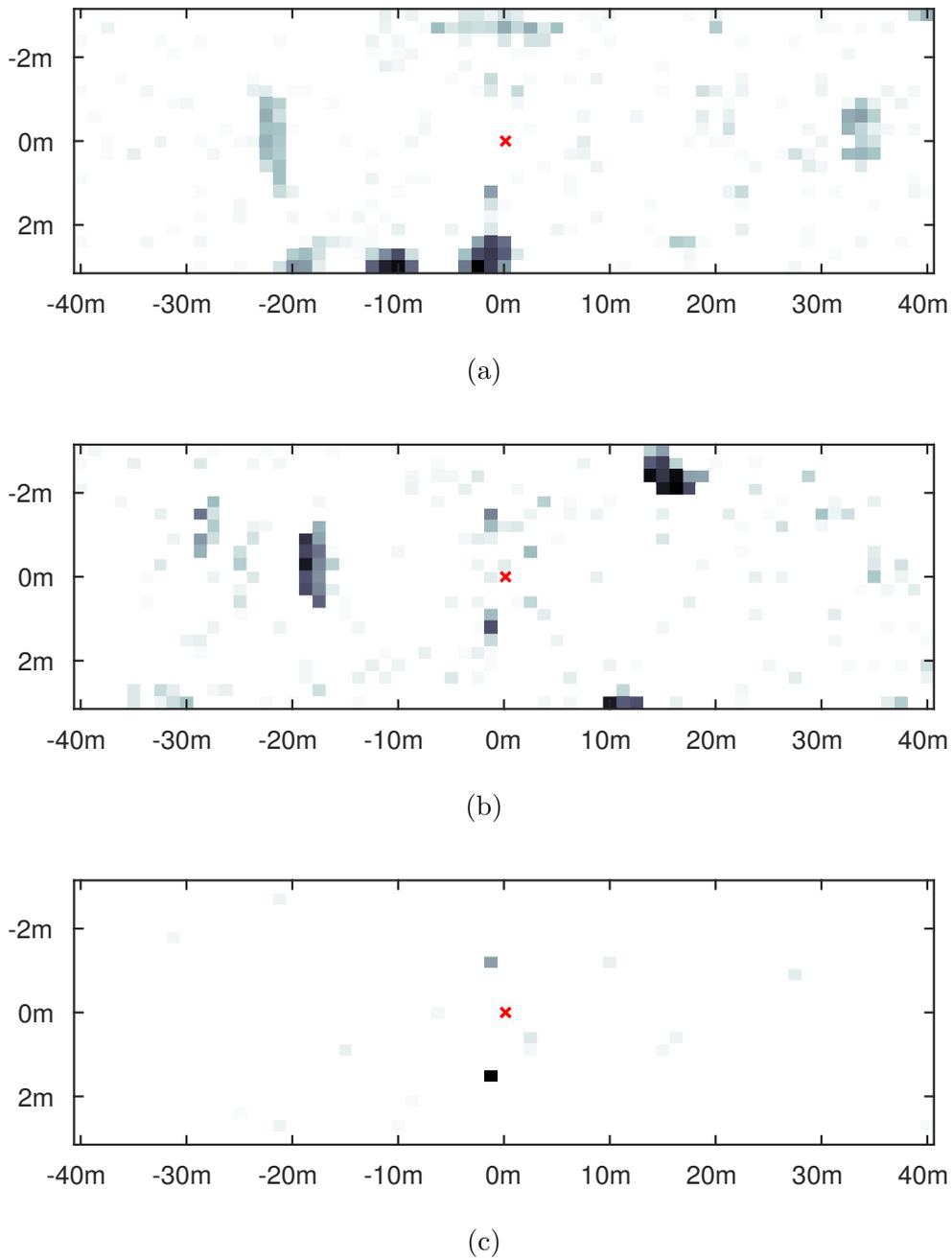


Figure 5.12. Features estimated from KITTI Scene 20. Shown are 3 out of 17 inferred features and the host vehicle (red cross). The first feature (a) indicates a lane change (right). The scene is rotated, showing two vehicles behind the host vehicle in the lower left corner. The second feature (b) represents moving at constant speed, most likely due to high traffic density (vehicle behind the host and on the left lane). The third feature (c) indicates acceleration and shows a free lane.

poor prediction performance. As proposed in Section 5.7.3, reweighting the action likelihood, assuming an action variable for each entry of the observation, yields a significant performance increase, while the observed states can still be reliably reconstructed.

An advantage of the proposed generative model is that it can be modified and extended easily. This is helpful especially for a different assumption on the feature weights. For example, if real-valued features are expected, the corresponding prior can be changed to a Gaussian distribution (c.f. Appendix B.1.3). Of course, inference has to be adapted accordingly. Further, one can easily extend the proposed model to a semi-supervised learning approach, in which we add variables for states where the actions are not observed. Thus, the state representations can be learned from even more data resulting in more accurate estimates.

5.11 Conclusion

In this chapter, we have presented a framework for learning from demonstrations based on BNFL. This framework allows us to analyze the observed behavior and to predict actions for new states. A key assumption in the presented model is that the observations are composed of latent features. Further, each feature imposes its own policy and contributes to the decision of the agent. To learn the structure of the behavior, we have proposed a Bayesian nonparametric approach based on the Indian Buffet Process, which allows to infer the number of features and the features itself from the observed data. By means of this model, we are able to obtain a deeper understanding of the observed behavior as the features and their policies allow to reason about the observed decisions. The simulations show that the developed algorithm performs well. Only in scenarios with strong noise and especially when the number of latent features is high with respect to the dimension of the observations, inference becomes challenging due to the fact that the algorithm finds several explanations for the observed data. To investigate the performance of the proposed algorithm on a real-world problem, we have considered the task of learning a driver's behavior. The learned features clearly indicate the reasons for the observed actions of the driver. Further, predicting actions for new observations of a hold-out data set demonstrates that actions can be predicted reliably.

Chapter 6

Conclusions and Outlook

In this thesis, we have proposed techniques to learn low-dimensional representations for the analysis of hyperspectral images and decision-making problems. A focus has been put on finding means to infer the dimension of the low-dimensional representations. For this, we have considered subspace-based methods as well as Bayesian nonparametric approaches.

In particular, we have presented a band-selection method and a sparse acquisition approach for classification in hyperspectral imaging, as well as an unmixing algorithm, revealing the latent endmembers and abundances. For the latter, we have considered a Bayesian nonparametric approach based on Bayesian Nonparametric Feature Learning (BNFL), expressing the belief over the realizations of the latent variables with respect to the observed data and the proposed model. A significant advantage of this approach, in comparison to existing methods, is that the number of endmembers, the actual endmembers, and their abundances can be jointly inferred from the observations.

Further, we have extended the BNFL approach for use in decision-making problems. We have considered a Learning From Demonstrations (LFD) problem, in which we assume that the agent makes its decision based on latent features. As each feature imposes its own policy, learning the features helps to reason about the observed behavior. Using the proposed framework, the features as well as the policy can be inferred from the observations. This enabling to understand the structure of the observations and to predict actions for new states.

In Section 6.1, we summarize our findings and draw conclusions about the problems considered in this thesis. An outlook regarding potential future work is given in Section 6.2.

6.1 Summary and Conclusions

6.1.1 Feature Learning for Classification in Hyperspectral Imaging

The proposed feature selection approach, designed for the use in hyperspectral imaging, addresses the problem of classifying high dimensional data. The goal is to decrease the

computational load during prediction and improve classification accuracy by overcoming the problems described by the curse of dimensionality. The presented approach, Spectral Clustering-Based Band Selection (SCBS), is based on self-tuning spectral clustering and groups similar bands into clusters. From each cluster, representatives are then chosen by means of PCA, which can be used for further processing, e.g., classification. We have shown based on three real data sets that the proposed scheme is able to outperform state-of-the-art methods and achieves comparably high classification accuracies especially for low number of bands. Using Support Vector Machine (SVM)s, we have obtained the highest accuracies in comparison to the tested methods, close to the results of considering all bands.

The presented low-dimensional acquisition method aims at only capturing information which is relevant for classification. Information, which is literally thrown away in feature selection, is ideally not even captured, making acquisition and evaluation of the image data more efficient. For this, we have extended the work on Compressive Sensing (CS) for Hyperspectral Imaging (HSI) to a Compressive Classification (CC) framework. Using the proposed approach, a computationally expensive reconstruction of the data is not required as in contrast to CS. Moreover, the measurement and transformation matrices are optimized using the available training data, improving classification accuracy significantly. The results based on real data show that the presented approach works well and easily outperforms existing, non-adaptive CC methods, yielding results close to conventional data acquisition and analysis.

6.1.2 Hyperspectral Unmixing via Bayesian Nonparametric Feature Learning

To learn the structure of a hyperspectral image, we have adapted the BNFL model and proposed a Bayesian nonparametric unmixing algorithm, revealing the endmembers and their abundances of the hyperspectral image. The endmembers are modeled by means of an Indian Buffet Process (IBP), which allows to infer the number of endmembers from the observations. Thus, the proposed algorithm is able to jointly estimate the endmembers, their fractional abundances, and, in contrast to most existing algorithms, the number of endmembers. The developed inference algorithm is based on Gibbs sampling. Due to the high flexibility of the model, the sampler might get trapped in a mode of the posterior. To solve this issue, we have proposed to make use of parallel tempering. Simulations with real spectra have shown that the proposed approach is able to accurately estimate the abundances and endmembers, comparable

to state-of-the-art algorithms, while simultaneously estimating the number of endmembers. Further, the performance is analyzed with real data experiments, revealing that the algorithm is able to produce accurate results which are in line with those reported in previous work. Only in case of strong noise, the number of endmembers is slightly overestimated. This effect is likely due to the fact that the model cannot explain the observed noise. Therefore, the noise is absorbed into few additional endmembers, which are basically noisy versions of existing endmembers.

6.1.3 Feature-Based Decision-making

While traditional LFD is concerned with the inference of the policy based on an observed behavior, we have proposed a model that is also able to explain the behavior by means of features. For this, we assume that the agent makes its decision based on latent features of the states, where each feature indicates a certain policy. Thus, analyzing the features can help to gain a deeper understanding of the observed behavior. In particular, we have extended the BNFL framework for the use in LFD, allowing to infer the number of features, the actual features, and their policies. The simulation experiments show that the developed algorithm estimates the features and the policies reliably. Only in scenarios with strong noise and many latent features, inference becomes challenging. To demonstrate the performance of the proposed algorithm, we have considered the task of learning a driver's behavior. For this, we have applied our algorithm to real data, obtained from the KITTI benchmark suite. The results reveal the conditions under which the driver takes the observed actions. Further, prediction on a hold-out data set demonstrates that actions can be accurately predicted.

6.2 Outlook

6.2.1 Low-Dimensional Representations for Classification in Hyperspectral Imaging

Exploiting the spatial information of neighboring pixels in band selection can help to reduce noise in the image and suppress outliers, yielding superior classification performances. This has been shown by the tests using morphological filtering. More advanced approaches that group similar pixels, e.g., based on Markov Random Fields (MRFs) are likely to result in even better features.

The Compressive Classification framework can be extended to use information about the classes, which we have assumed to be available, for learning the measurement and basis matrices. This leads to a new formulation of the problem, where the classification error instead of the loss of the reconstructions is minimized. A solution to this problem could be based on a Deep Learning (DL) approach, in which the first layer of the network is linear, representing the measurement matrix. During data acquisition, this layer could be decoupled from the network and represented by means of a Digital Mirror Device (DMD). However, this approach requires a deeper analysis of the network and its properties. Especially incorporating the Restricted Isometry Property (RIP) constraint can be challenging. Generally, minimizing the classification error by optimizing the matrices is likely to be computational demanding and sophisticated approaches are required to alleviate the computational requirements.

Of course, there are other possibilities to learn features for classification, especially in hyperspectral imaging. As mentioned in the introduction, DL has shown promising results in many classification tasks. Thus, applying deep architectures to hyperspectral image classification is an interesting research area.

6.2.2 Features for the Analysis of Hyperspectral Images

For the Bayesian Nonparametric Unmixing, we have derived a Gibbs sampler that is able to accurately sample from the posterior distribution. However, a significant drawback of this approach is the runtime of the proposed algorithm. If the number of endmembers is known *a priori*, an alternative sampling method can be used, e.g., the Metropolis-Hamilton algorithm which often provides less correlated samples than the Gibbs sampler. Thus, this approach requires less sampling steps to represent the posterior distribution. Alternatively, variational methods can be investigated. Though variational approximations may yield poor results using an IBP, the relation between the IBP and the Beta process may provide new opportunities to perform variational inference.

A different interesting direction is to extend this framework to a semi-supervised approach. The semi-supervised approach could be based on different assumptions on the available data: (i) the abundances of some pixels are known, (ii) some of the endmembers are known, or (iii) the abundances of some pixels and some endmembers are known. The proposed model can be extended to exploit any of the three mentioned assumptions. The additional information can be used to guide the unmixing algorithm, providing more accurate unmixing results.

As unmixing reveals the pure materials present in the scene and their fractional abundances, this information can be used to reconstruct a high-resolution image of the scene. A super-resolution algorithm could proceed as follows: At first, the resolution of the new image would have to be set. By means of a probabilistic model, the probability of each high-resolution pixel belonging to a certain class (endmember) would then be estimated based on the abundances of the endmembers.

6.2.3 Features for the Analysis and Prediction in Decision-Making Problems

As the Bayesian nonparametric LFD and the Bayesian nonparametric unmixing approaches are based on the same model, the outlook given in the previous section mostly applies also to the model for LFD. As detailed in Section 6.2.2, different sampling approaches can be considered to speed up the inference process.

A general problem in supervised learning is that labeled data is usually expensive to obtain. Thus, the parameters are often learned from small data sets only, resulting in high uncertainty about the variables that shall be inferred. This problem also concerns the proposed LFD approach. However, the proposed model can be easily extended to a semi-supervised framework by also considering state variables with latent actions. Thus, learning is based on more observations, yielding more accurate estimates.

As explained, for the analysis of the observed behavior, the transition model mainly provides additional information which helps to obtain more accurate estimates. In contrast, if the inferred policy shall be used to determine the behavior of an agent, considering the model is crucial. Only the model can guarantee that tractable action-state transitions are exploited when learning an optimal behavior. Thus, incorporating the transition model in the proposed framework and developing an efficient inference scheme is another interesting future direction.

Appendix A

Derivations for Compressive Classification

The energy terms enforcing sparsity and orthogonality are given as

$$E_{\text{Sparse}}(\Psi, \mathbf{X}) = \sum_{n=1}^N \left\| \mathbf{z}_n - \sum_{k=1}^K \frac{\psi_k}{\|\psi_k\|_2} x_{k,n} \right\|_2^2 + b_{\text{Sparse}} \sum_{n=1}^N \|\mathbf{x}_n\|_1.$$

$$E_{\text{Orth}}(\Psi, \mathbf{X}) = b_{\text{Orth}} \left\| \sum_{k=1}^K \left(\frac{\psi_k}{\|\psi_k\|_2} \right)^T \frac{\psi_k}{\|\psi_k\|_2} - \mathbf{I} \right\|_2^2,$$

As we want to optimize both terms with respect to the basis matrix, Ψ , we need to find expression for the derivatives. Using Eqs. (87), (88), and (130) in [213], yields,

$$\begin{aligned} \frac{\partial E_{\text{Sparse}}(\Psi, \mathbf{X})}{\partial \psi_k} &= -2 \left(\mathbf{z}_n - \sum_{k'=1}^K \frac{\psi_{k'}}{\|\psi_{k'}\|_2} x_{k',n} \right) x_{k,n} \left(\frac{\mathbf{I}}{\|\psi_k\|_2} - \frac{\psi_k \psi_k^T}{\|\psi_k\|_2^3} \right)^T \\ &= 2 \frac{x_{k,n}}{\|\psi_k\|_2} \left(-\mathbf{z}_n + \sum_{k'=1}^K x_{k',n} \frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right. \\ &\quad \left. + \mathbf{z}_n \frac{(\psi_k \psi_k^T)^T}{\|\psi_k\|_2^2} - \frac{(\psi_k \psi_k^T)^T}{\|\psi_k\|_2^2} \sum_{k'=1}^K x_{k',n} \frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right) \end{aligned}$$

For the orthogonal constraint, we obtain

$$\begin{aligned} \frac{\partial E_{\text{Orth}}(\Psi, \mathbf{X})}{\partial \psi_k} &= 2b_{\text{Orth}} \left(\sum_{k'=1}^K \left(\frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right)^T \frac{\psi_{k'}}{\|\psi_{k'}\|_2} - \mathbf{I} \right) \\ &\quad \times 2 \left(\sum_{k'=1}^K \frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right) \left(\frac{\mathbf{I}}{\|\psi_k\|_2} - \frac{\psi_k \psi_k^T}{\|\psi_k\|_2^3} \right)^T \\ &= b_{\text{Orth}} \frac{4}{\|\psi_k\|_2} \left(\sum_{k'=1}^K \frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right) \left(\sum_{k'=1}^K \left(\frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right)^T \frac{\psi_{k'}}{\|\psi_{k'}\|_2} - \mathbf{I} \right) \\ &\quad - \frac{(\psi_k \psi_k^T)^T}{\|\psi_k\|_2^2} \sum_{k'=1}^K \left(\frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right)^T \frac{\psi_{k'}}{\|\psi_{k'}\|_2} + \frac{(\psi_k \psi_k^T)^T}{\|\psi_k\|_2^2} \end{aligned}$$

Thus, the positive and negative terms of the derivative are given as

$$\begin{aligned}
\left[\frac{\partial E(\Psi, \mathbf{X})}{\partial \psi_k} \right]^+ &= 2 \frac{x_{k,n}}{\|\psi_k\|_2} \left(\sum_{k'=1}^K x_{k',n} \frac{\psi_{k'}}{\|\psi_{k'}\|_2} + \mathbf{z}_n \frac{(\psi_k \psi_k^T)^T}{\|\psi_k\|_2^2} \right) \\
&+ \frac{4b_{\text{Orth}}}{\|\psi_k\|_2} \left(\sum_{k'=1}^K \frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right) \left(\sum_{k'=1}^K \left(\frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right)^T \frac{\psi_{k'}}{\|\psi_{k'}\|_2} + \frac{(\psi_k \psi_k^T)^T}{\|\psi_k\|_2^2} \right) \\
&= 2 \frac{x_{k,n}}{\|\psi_k\|_2} \left(\sum_{k'=1}^K x_{k',n} \tilde{\psi}_{k'} + \mathbf{z}_n \left(\tilde{\psi}_k \tilde{\psi}_k^T \right)^T \right) \\
&+ \frac{4b_{\text{Orth}}}{\|\psi_k\|_2} \left(\sum_{k'=1}^K \tilde{\psi}_{k'} \right) \left(\sum_{k'=1}^K \tilde{\psi}_{k'}^T \tilde{\psi}_{k'} + \left(\tilde{\psi}_k \tilde{\psi}_k^T \right)^T \right) \\
\left[\frac{\partial E(\Psi, \mathbf{X})}{\partial \psi_k} \right]^- &= 2 \frac{x_{k,n}}{\|\psi_k\|_2} \left(\mathbf{z}_n + \frac{(\psi_k \psi_k^T)^T}{\|\psi_k\|_2^2} \sum_{k'=1}^K x_{k',n} \frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right) \\
&+ \frac{4b_{\text{Orth}}}{\|\psi_k\|_2} \left(\sum_{k'=1}^K \frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right) \left(1 + \frac{(\psi_k \psi_k^T)^T}{\|\psi_k\|_2^2} \sum_{k'=1}^K \left(\frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right)^T \frac{\psi_{k'}}{\|\psi_{k'}\|_2} \right) \\
&= 2 \frac{x_{k,n}}{\|\psi_k\|_2} \left(\mathbf{z}_n + \left(\tilde{\psi}_k \tilde{\psi}_k^T \right)^T \sum_{k'=1}^K x_{k',n} \tilde{\psi}_{k'} \right) \\
&+ \frac{4b_{\text{Orth}}}{\|\psi_k\|_2} \left(\sum_{k'=1}^K \tilde{\psi}_{k'} \right) \left(1 + \left(\tilde{\psi}_k \tilde{\psi}_k^T \right)^T \sum_{k'=1}^K \tilde{\psi}_{k'}^T \tilde{\psi}_{k'} \right)
\end{aligned}$$

Appendix B

Derivations for Bayesian Nonparametric Feature Learning

B.1 Conditionals for the Feature Weights

The conditional of the feature weights depends on the observation likelihood and the weight prior in the proposed models,

$$p(\mathbf{w}_k | -) \propto p(\mathbf{Z} | \mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z^2) p(\mathbf{w}_k | \mathbf{W} \setminus \mathbf{w}_k, \theta_w),$$

where θ_w denotes the set of hyperparameters and $\mathbf{W} \setminus \mathbf{w}_k$ denotes the feature weight matrix without the k th column. Since

$$p(\mathbf{w}_k | \mathbf{W} \setminus \mathbf{w}_k, \theta_w) \propto p(\mathbf{W} | \theta_w),$$

the conditional is always of the following form,

$$\begin{aligned} p(\mathbf{w}_k | -) &\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \left(\mathbf{z}_n - \sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right)^T \left(\mathbf{z}_n - \sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right)\right. \\ &\quad \left. + \log p(\mathbf{W} | \theta_w)\right\} \\ &\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \left(-2\mathbf{z}_n^T \underbrace{\sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}}_{\propto (\mathbf{a}_k \odot \mathbf{w}_k) s_{k,n}}\right.\right. \\ &\quad \left. \left. + \underbrace{\left(\sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right)^T \left(\sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right)}_{\propto (\mathbf{a}_k \odot \mathbf{w}_k)^T (\mathbf{a}_k \odot \mathbf{w}_k) s_{k,n}^2 + 2(\mathbf{a}_k \odot \mathbf{w}_k) s_{k,n} \sum_{\substack{k'=1 \\ k' \neq k}}^K (\cdot)}\right) + \log p(\mathbf{W} | \theta_w)\right\} \\ &\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \left(-2s_{k,n} \mathbf{z}_n^T (\mathbf{a}_k \odot \mathbf{w}_k) + s_{k,n}^2 (\mathbf{a}_k \odot \mathbf{w}_k)^T (\mathbf{a}_k \odot \mathbf{w}_k)\right.\right. \\ &\quad \left. \left. + 2s_{k,n} (\mathbf{a}_k \odot \mathbf{w}_k)^T \sum_{\substack{k'=1 \\ k' \neq k}}^K s_{k',n} (\mathbf{a}_{k'} \odot \mathbf{w}_{k'})\right) + \log p(\mathbf{W} | \theta_w)\right\} \end{aligned}$$

B.1.1 Using the Distance Prior

The conditional using the Distance prior (c.f. supplementary material for [134]) is derived as follows,

$$\begin{aligned}
p(\mathbf{w}_k | -) &\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \left(\mathbf{z}_n - \sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right)^T \left(\mathbf{z}_n - \sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right) \right. \\
&\quad \left. - \gamma \sum_{k'=1}^K \left(\mathbf{w}_{k'} - \frac{1}{K} \sum_{k''=1}^K \mathbf{w}_{k''}\right)^T \left(\mathbf{w}_{k'} - \frac{1}{K} \sum_{k''=1}^K \mathbf{w}_{k''}\right)\right\} \\
&\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \left(\mathbf{z}_n - \sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right)^T \left(\mathbf{z}_n - \sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right) \right. \\
&\quad \left. - \gamma \left(\underbrace{\sum_{k'=1}^K \mathbf{w}_{k'}^T \mathbf{w}_{k'}}_{\propto \mathbf{w}_k^T \mathbf{w}_k} - 2 \underbrace{\sum_{k'=1}^K \mathbf{w}_{k'}^T \frac{1}{K} \sum_{k''=1}^K \mathbf{w}_{k''}}_{\frac{1}{K} \mathbf{w}_k^T \left(\mathbf{w}_k + 2 \sum_{\substack{k'=1 \\ k' \neq k}}^K \mathbf{w}_{k'}\right)} + \sum_{k'=1}^K \frac{1}{K^2} \left(\sum_{k''=1}^K \mathbf{w}_{k''}\right)^T \left(\sum_{k''=1}^K \mathbf{w}_{k''}\right) \right) \right\} \\
&\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \left(-2s_{k,n} \mathbf{z}_n^T (\mathbf{a}_k \odot \mathbf{w}_k) + s_{k,n}^2 (\mathbf{a}_k \odot \mathbf{w}_k)^T (\mathbf{a}_k \odot \mathbf{w}_k) \right. \right. \\
&\quad \left. \left. + 2s_{k,n} (\mathbf{a}_k \odot \mathbf{w}_k)^T \sum_{\substack{k'=1 \\ k' \neq k}}^K s_{k',n} (\mathbf{a}_{k'} \odot \mathbf{w}_{k'})\right) \right. \\
&\quad \left. - \gamma \left(\underbrace{\mathbf{w}_k^T \mathbf{w}_k - 2 \frac{1}{K} \sum_{k'=1}^K \mathbf{w}_{k'}^T \sum_{k''=1}^K \mathbf{w}_{k''} + \frac{1}{K} \sum_{k''=1}^K \mathbf{w}_{k''}^T \sum_{k''=1}^K \mathbf{w}_{k''}}_{-\frac{1}{K} \mathbf{w}_k^T \left(\mathbf{w}_k + 2 \sum_{\substack{k'=1 \\ k' \neq k}}^K \mathbf{w}_{k'}\right)} \right) \right\} \\
&\propto \exp\left\{-\frac{1}{2} \mathbf{w}_k^T \mathbf{A} \mathbf{w}_k + \mathbf{b}^T \mathbf{w}_k\right\},
\end{aligned}$$

with

$$\mathbf{A} = \left(\frac{1}{\sigma_z^2} \sum_{n=1}^{N_z} s_{k,n}^2 \right) \text{diag}(\mathbf{a}_k) + 2\gamma \left(1 - \frac{1}{K}\right) \mathbf{I},$$

and

$$\mathbf{b} = \frac{1}{\sigma_z^2} \sum_{n=1}^{N_z} s_{k,n} \left(\mathbf{z}_n - \sum_{\substack{k'=1 \\ k' \neq k}}^K s_{k',n} (\mathbf{a}_{k'} \odot \mathbf{w}_{k'})\right) \odot \mathbf{a}_k + \gamma \frac{2}{K} \sum_{\substack{k'=1 \\ k' \neq k}}^K \mathbf{w}_{k'}.$$

Hence,

$$\begin{aligned}
\Sigma &= \mathbf{A}^{-1}, \\
\boldsymbol{\mu} &= \mathbf{A}^{-1} \mathbf{b}.
\end{aligned}$$

Note that the conditional, $p(\mathbf{w}_k | -)$, is a truncated multivariate Gaussian distribution with covariance matrix Σ and mean $\boldsymbol{\mu}$, where the elements of \mathbf{w}_k are constrained to be positive-valued.

B.1.2 Using an Exponential Prior

With an Exponential prior on the feature weights, \mathbf{W} , we obtain the following conditional distribution for the columns of \mathbf{W} ,

$$\begin{aligned}
p(\mathbf{w}_k | -) &\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \left(\mathbf{z}_n - \sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right)^T \left(\mathbf{z}_n - \sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right)\right. \\
&\quad \left. - \frac{1}{\gamma_w} \sum_{k'=1}^K \mathbf{w}_{k'}\right\} \\
&\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \left(-2s_{k,n} \mathbf{z}_n^T (\mathbf{a}_k \odot \mathbf{w}_k) + s_{k,n}^2 (\mathbf{a}_k \odot \mathbf{w}_k)^T (\mathbf{a}_k \odot \mathbf{w}_k)\right.\right. \\
&\quad \left.\left. + 2s_{k,n} (\mathbf{a}_k \odot \mathbf{w}_k)^T \sum_{\substack{k'=1 \\ k' \neq k}}^K s_{k',n} (\mathbf{a}_{k'} \odot \mathbf{w}_{k'})\right) - \frac{1}{\gamma_w} \mathbf{w}_k\right\} \\
&\propto \exp\left\{-\frac{1}{2} \mathbf{w}_k^T \mathbf{A} \mathbf{w}_k + \mathbf{b}^T \mathbf{w}_k\right\},
\end{aligned}$$

with

$$\mathbf{A} = \left(\frac{1}{\sigma_z^2} \sum_{n=1}^{N_z} s_{k,n}^2\right) \text{diag}(\mathbf{a}_k)$$

and

$$\mathbf{b} = \frac{1}{\sigma_z^2} \sum_{n=1}^{N_z} s_{k,n} \left(\mathbf{z}_n - \sum_{\substack{k'=1 \\ k' \neq k}}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right) \odot \mathbf{a}_k - \frac{1}{\gamma_w}.$$

Hence,

$$\begin{aligned}
\Sigma &= \mathbf{A}^{-1}, \\
\boldsymbol{\mu} &= \mathbf{A}^{-1} \mathbf{b}.
\end{aligned}$$

Note that the conditional, $p(\mathbf{w}_k | -)$, is a truncated multivariate Gaussian distribution with covariance matrix Σ and mean $\boldsymbol{\mu}$, where the elements of \mathbf{w}_k are constrained to be positive-valued.

B.1.3 Using a Gaussian Prior

With a Gaussian prior on the feature weights,

$$p(\mathbf{W} | \sigma_w^2) = \prod_{k=1}^K \prod_{d=1}^D \mathcal{N}_{w_{d,k}}(0, \sigma_w^2),$$

we obtain the following conditional distribution for the columns of \mathbf{W} ,

$$\begin{aligned} p(\mathbf{w}_k | -) &\propto p(\mathbf{Z} | \mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z^2) p(\mathbf{W} | \sigma_w^2) \\ &\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \left(\mathbf{z}_n - \sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right)^T \left(\mathbf{z}_n - \sum_{k'=1}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right)\right. \\ &\quad \left. - \frac{1}{2\sigma_w^2} \sum_{k'=1}^K \mathbf{w}_{k'}^T \mathbf{w}_{k'}\right\} \\ &\propto \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} (-2s_{k,n} \mathbf{z}_n^T (\mathbf{a}_k \odot \mathbf{w}_k) + s_{k,n}^2 (\mathbf{a}_k \odot \mathbf{w}_k)^T (\mathbf{a}_k \odot \mathbf{w}_k)\right. \\ &\quad \left. - \frac{1}{2\sigma_w^2} \mathbf{w}_k^T \mathbf{w}_k\right\} \\ &\propto \exp\left\{-\frac{1}{2} \mathbf{w}_k^T \mathbf{A} \mathbf{w}_k + \mathbf{b}^T \mathbf{w}_k\right\}, \end{aligned}$$

with

$$\mathbf{A} = \left(\frac{1}{\sigma_z^2} \sum_{n=1}^{N_z} s_{k,n}^2\right) \text{diag}(\mathbf{a}_k) + \frac{1}{\sigma_w^2} \mathbf{I}$$

and

$$\mathbf{b} = \frac{1}{\sigma_z^2} \sum_{n=1}^{N_z} s_{k,n} \left(\mathbf{z}_n - \sum_{\substack{k'=1 \\ k' \neq k}}^K (\mathbf{a}_{k'} \odot \mathbf{w}_{k'}) s_{k',n}\right) \odot \mathbf{a}_k.$$

Hence,

$$\begin{aligned} \boldsymbol{\Sigma} &= \mathbf{A}^{-1}, \\ \boldsymbol{\mu} &= \mathbf{A}^{-1} \mathbf{b}. \end{aligned}$$

Note that the conditional, $p(\mathbf{w}_k | -)$, is multivariate Gaussian distribution with covariance matrix $\boldsymbol{\Sigma}$ and mean $\boldsymbol{\mu}$.

B.2 Conditional for the Noise Variance

Since the variance, σ_z^2 , of the observation likelihood is Inverse-Gamma distributed with hyperparameters α_σ and β_σ , we obtain

$$\begin{aligned}
p(\sigma_z^2 | -) &\propto P(\mathbf{Z} | \mathbf{W}, \mathbf{A}, \mathbf{S}, \sigma_z^2) P(\sigma_z^2 | \alpha_\sigma, \beta_\sigma) \\
&\propto (\sigma_z^2)^{-ND/2} \exp\left\{-\frac{1}{2\sigma_z^2} \sum_{n=1}^{N_z} \sum_{d=1}^D \left(z_{d,n} - \sum_{k=1}^K a_{d,k} w_{d,k} s_{k,n}\right)^2\right\} (\sigma_z^2)^{-\alpha_\sigma-1} \exp\left\{-\frac{\beta_\sigma}{\sigma_z^2}\right\} \\
&\propto (\sigma_z^2)^{-ND/2-\alpha_\sigma-1} \exp\left\{-\left(\frac{1}{2} \sum_{n=1}^{N_z} \sum_{d=1}^D \left(z_{d,n} - \sum_{k=1}^K a_{d,k} w_{d,k} s_{k,n}\right)^2 - \beta_\sigma\right) \frac{1}{\sigma_z^2}\right\} \\
&\propto \text{IGa}_{\sigma_z^2} \left(\frac{ND}{2} + \alpha_\sigma, \frac{1}{2} \sum_{n=1}^{N_z} \sum_{d=1}^D \left(z_{d,n} - \sum_{k=1}^K a_{d,k} w_{d,k} s_{k,n}\right)^2 + \beta_\sigma\right).
\end{aligned}$$

B.3 Sparsity-Promoting Prior on the Feature Coefficients

In Chapter 5, we assume a mixture model promoting sparsity on the substates, similar to a *spike* and *slab* prior. We consider independence of the features and place a Beta prior over the mixture weights, θ_0 and θ_1 . Note that $\theta_1 = 1 - \theta_0$, simplifying marginalization,

$$\begin{aligned}
P(\mathbf{S}) &= \prod_{k=1}^K \int_0^1 \prod_{n=1}^N \left(P(s_{n,k} = 0 | \theta_0) \delta(s_{n,k}) + P(s_{n,k} \neq 0 | \theta_0) \delta_{\neq 0}(s_{n,k}) \right) p(\theta_0) d\theta_0 \\
&\propto \prod_{k=1}^K \int_0^1 \prod_{n=1}^N \left(\theta_0^{\delta(s_{n,k}) + \alpha_{s0}} \theta_1^{\alpha_{s1}} \delta(s_{n,k}) + \theta_1^{\delta_{\neq 0}(s_{n,k}) + \alpha_{s1}} \theta_0^{\alpha_{s0}} \frac{1}{L-1} \delta_{\neq 0}(s_{n,k}) \right) d\theta_0 \\
&\propto \prod_{k=1}^K \int_0^1 \left(\theta_0^{\sum_{n=1}^N \delta(s_{n,k}) + \alpha_{s0}} \theta_1^{\alpha_{s1}} \theta_1^{\sum_{n=1}^N \delta_{\neq 0}(s_{n,k}) + \alpha_{s1}} \theta_0^{\alpha_{s0}} \frac{1}{L-1} \right) d\theta_0 \\
&\propto \prod_{k=1}^K \int_0^1 \left(\theta_0^{m_{s=0,k} + 2\alpha_{s0}} \theta_1^{m_{s \neq 0,k} + 2\alpha_{s1}} \frac{1}{L-1} \right) d\theta_0 \\
&\propto \prod_{k=1}^K \text{BetaBin}_{s_k}(m_{s=0,k} + 2\alpha_{s0}, m_{s \neq 0,k} + 2\alpha_{s1}),
\end{aligned}$$

where $\delta_{\neq 0}(s)$ returns 1 if $s \neq 0$ and 0 otherwise and $m_{s=0,k} = \sum_{n=1}^N \delta(s_{n,k})$, $m_{s \neq 0,k} = \sum_{n=1}^N \delta_{\neq 0}(s_{n,k})$. Sampling from the resulting Beta-Binomial distribution

is straightforward using a Gibbs sampler as the conditional is given as

$$P(s_{n,k} = 0 | -) \propto m_{s=0,k} + 2\alpha_{s0},$$

and

$$P(s_{n,k} \neq 0 | -) \propto m_{s \neq 0,k} + 2\alpha_{s1},$$

Note that $s_{n,k}$ follows a Categorical distribution.

List of Acronyms

ACC	Adaptive Compressed Classification
AIC	Akaike Information Criterion
AP	Affinity Propagation
AVIRIS	Airborne Visible/Infrared Imaging Spectrometer
BFL	Bayesian Feature Learning
BLU	Bayesian Linear Unmixing
BNFL	Bayesian Nonparametric Feature Learning
BNU	Bayesian Nonparametric Unmixing
BSS	Blind Source Separation
CA	Class Accuracy
CBS	Constrained Band Selection
CC	Compressive Classification
CFA	Contingent Feature Analysis
CM	Contrast Measure
CP	Center of Pavia
CRP	Chinese Restaurant Process
CS	Compressive Sensing
DAIS	Digital Airborne Imaging Spectrometer
DAG	Directed Acyclic Graph
DL	Deep Learning
DLR	<i>Deutsches Zentrum für Luft- und Raumfahrt</i>
DMD	Digital Mirror Device
ELM	Eigenvalue Likelihood Maximization
EM	Expectation Maximization

FL	Feature Learning
FMDP	Factored Markov Decision Process
HideNN	Denoised Hyperspectral Intrinsic Dimensionality Estimation With Nearest-Neighbor Distance Ratios
HSI	Hyperspectral Imaging
HSU	Hyperspectral Unmixing
HySime	Hyperspectral Signal Subspace Identification by Minimum Error
IBP	Indian Buffet Process
ICA	Independent Component Analysis
ICE	Iterated Constrained Endmembers
IP	Indiana's Indian Pines
IRL	Inverse Reinforcement Learning
KNN	k -Nearest Neighbor
LCMV	Linearly Constrained Minimum Variance
LFD	Learning From Demonstrations
LLC	Local Coordinate Coding
MAD	Mean Absolute Deviation
MAP	<i>maximum-a-posteriori</i>
MCMC	Markov Chain Monte Carlo
MDL	Minimum Description Length
MDP	Markov Decision Process
MH	Metropolis-Hastings
MMSE	Minimum Mean Squared Error
MRF	Markov Random Field
MSE	Mean Squared Error

MVES	Minimum-Volume Enclosing Simplex
MVPCA	Maximum-Variance PCA
MVT	Minimum Volume Transform
NMF	Non-negative Matrix Factorization
OA	Overall Accuracy
PCA	Principal Component Analysis
pdf	probability density function
POMDP	Partially Observable Markov Decision Process
PPI	Pixel Purity Index
PT	Parallel Tempering
RBF	Radial Basis Function
RL	Reinforcement Learning
RMSE	Root Mean Squared Error
RIP	Restricted Isometry Property
RODIS	Reflective Optics System Imaging Spectrometer
SC	Spectral Clustering
SCBS	Spectral Clustering-Based Band Selection
SID	Spectral Information Divergence
SF	Smashed Filter
SFA	Slow Feature Analysis
SNR	Signal-to-Noise Ratio
SPICE	Sparsity-Promoting ICE
SVM	Support Vector Machine
UP	University of Pavia
VCA	Vertex Component Analysis
VD	Virtual Dimensionality

List of Symbols

Symbols

$\mathbf{0}$	Null vector
\mathbb{R}	Set of real numbers
$\mathbb{R}_{+\setminus 0}$	Set of positive-valued real numbers
\mathbb{R}_+	Set of positive-valued real numbers including zero
α_γ	Parameter for the prior of γ_w
α_σ	Parameter for the prior of σ_z^2
α_ϕ	Parameter for the prior of ϕ
α_a	Sparsity parameter of the IBP
α_{IC}	Threshold on the information content
α_s	Parameter for the prior of \mathbf{s}
$\alpha_{s=0}$	Parameter for the prior of s if $s = \check{s}_1$
$\alpha_{s\neq 0}$	Parameter for the prior of s if $s \neq \check{s}_1$
α_{SVM}	Lagrangian multiplier (SVM)
$\boldsymbol{\alpha}_{SVM}$	Vector containing the Lagrangian multipliers α_{SVM}
β_γ	Parameter for the prior of γ_w
β_σ	Parameter for the prior of σ_z^2
β_a	Parameter for the IBP
β_a'	Proposed hyperparameter for the IBP
β_{IP}	Parameter of the illumination perturbation
γ_w	Parameter for the prior of \mathbf{W}
γ_{RBF}	Bandwidth parameter of the RBF kernel
γ_{SC}	Bandwidth parameter of the affinity kernel function
ϵ	Threshold on the tolerated reconstruction error in CS
ϵ_Φ	RMSE of the policies
$\epsilon_{\mathbf{F}}$	RMSE of the endmembers
ϵ_K	RMSE of the number of endmembers
$\epsilon_{\mathbf{S}}$	RMSE of the abundances
ϵ_{SID}	RMSE of the SID

$\epsilon_{\mathbf{x}}$	RMSE of the states
θ	Set containing feature realizations
θ^+	Set containing new feature realizations
θ_a	Parameter of the prior for \mathbf{A}
$\theta_{s=0}$	Parameter for the prior of s if $s = \check{s}_1$
$\theta_{s \neq 0}$	Parameter for the prior of s if $s \neq \check{s}_1$
$\bar{\theta}_{\mathbf{F}}$	Average angular difference between true and estimated features
$\bar{\theta}_{\mathbf{s}}$	Average angular difference between true and estimated abundances
θ	Parameter/variable of the predictive function
λ	Eigenvalue
Λ	Diagonal matrix containing the eigenvalues
$\mu_{\mathbf{s}}$	Conditional mean of \mathbf{s}
$\mu_{\mathbf{w}}$	Conditional mean of \mathbf{w}
ξ	Slack variable
ξ	Vector of slack variables
σ_z	Standard deviation of the noise of the observations
σ_w	Standard deviation of the features
$\Sigma^{(k)}$	Covariance matrix of the bands in cluster k
$\Sigma_{\mathbf{s}}$	Conditional covariance matrix of \mathbf{s}
$\Sigma_{\mathbf{w}}$	Conditional covariance matrix of \mathbf{w}
ϕ	Measurement vector of Φ / feature policy
Φ	Measurements matrix in CS / feature policies
$\tilde{\Phi}$	Measurements matrix in CS (reduced)
$\hat{\Phi}$	Estimate of Φ
$\hat{\Phi}'$	Optimal $D \times D$ sensing matrix (low rank)
$\hat{\Phi}_{\text{Opt}}$	Optimal $D \times D$ sensing matrix (full rank)
Ψ	Sparse transform matrix for CS
$\hat{\Psi}$	Estimate of Ψ

$\tilde{\Psi}$	Sparse transform matrix for CS (reduced)
Ω	Set of latent variables
$\mathbf{\Omega}$	Joint space of all latent variables
Θ	Domain of $\boldsymbol{\theta}$
\mathcal{C}	Set of classes
\mathcal{D}	Data set
\mathcal{D}_T	Training data set
\mathcal{F}	Feature space
\mathcal{O}_s	State space
\mathcal{O}_u	Action space
\mathcal{O}_z	Observation space
\mathcal{R}	Region in which z resides
\mathcal{S}	Domain of the sparse coefficients / feature coefficients / sub-states
\mathcal{X}	Domain of the sparse coefficients
\mathcal{Y}	Domain of the predictions
\mathcal{Z}	Domain of the observations
$a_{d,k}$	Element of \mathbf{A} in row d and column k
$a_{d,k}^+$	Element of \mathbf{A}^+ in row d and column k
$a_{SC,n,m}$	Affinity between n th and m th observation
$\tilde{a}_{SC,n,m}$	Locally scaled affinity between n th and m th observation
\mathbf{a}_k	k th column of \mathbf{A}
$\mathbf{a}_{k \setminus d}$	k th column of \mathbf{A} without the d th element
$\mathbf{a}_{\infty,k}$	k th column of \mathbf{A}_{∞}
$\mathbf{a}_{\infty,k \setminus d}$	k th column of \mathbf{A}_{∞} without the d th element
A_u	Accuracy of the prediction of actions
$[\mathbf{A}]$	Class of feature activation matrices
$[\mathbf{A}_{\infty}]$	Class of infinite feature activation matrices
\mathbf{A}	Endmember/feature activation matrix
\mathbf{A}^+	Activation matrix of new features
\mathbf{A}_{∞}	Feature activation matrix with an infinite number of columns
\mathbf{A}_{SC}	Affinity matrix in SC

b_{Orth}	Orthogonality regularization parameter
b_{Sparse}	Sparsity regularization parameter
b_{SVM}	Bias of the hyperplane of an SVM
\hat{b}_{SVM}	Learned b_{SVM}
\mathbf{b}	Vectorized band of a hyperspectral image
$C^{(k)}$	k th cluster
C_{SVM}	Regularization of the slack variables
d_{HO}	Distance between host vehicle and obstacle
D	Length of an observation vector / number of bands of a hyperspectral image
$D^{(k)}$	Number of bands in the k th cluster
$\tilde{D}^{(k)}$	Number of bands in the k th transformed cluster
\mathbf{D}_K	K -dimensional diagonal matrix
\mathbf{D}_{SC}	Diagonal matrix where the values on the diagonal represent the respective sum over the column of \mathbf{A}_{SC}
\mathbf{e}	Noise vector
f_s	Sampling frequency
\mathbf{f}	Feature vector
$\check{\mathbf{f}}_k$	True endmember/feature vector
$\hat{\mathbf{f}}_k$	Estimated endmember/feature vector
\mathbf{F}	Endmember/feature matrix
$\hat{\mathbf{F}}$	Estimate of the endmember/feature matrix
$h_{\alpha_A}^{(1)}, h_{\alpha_A}^{(2)}$	Parameters for the prior of α_a
$h_{\alpha_\sigma}^{(1)}, h_{\alpha_\sigma}^{(2)}$	Parameters for the prior of α_σ
$h_{\beta_A}^{(1)}, h_{\beta_A}^{(2)}$	Parameters for the prior of β_a
$h_{\beta_\sigma}^{(1)}, h_{\beta_\sigma}^{(2)}$	Parameters for the prior of β_σ
I_h	Difference of the observations (unscaled)
I_l	Difference of the observations (scaled)
$\text{IC}^{(k)}$	Information content in cluster k
\mathbf{I}_K	K -dimensional identity matrix

K	Number of features/basis vectors/measurements in CS
K_+	Number of active features (nonzero columns of \mathbf{A}_∞)
K'_+	Number of new active features
\bar{K}_+	Expected number of active features
$K_{\mathbf{h}}$	Number of occurrences of binary vector \mathbf{h} in the columns of \mathbf{A}
K_l	Difference of the kernel functions for scaled observations
K_h	Difference of the kernel functions for unscaled observations
K_S	Number of non-zero elements in \mathbf{x}
K_{SC}	Number of clusters in SC
$K_{\mathcal{R}}$	Number of training samples in \mathcal{R}
L	Size of \mathcal{S}
L_S	Number of samples skipped in Metropolis and Metropolis-Hastings sampling
\mathbf{L}_{SC}	Normalized graph Laplacian
m_k	Sum over the elements of \mathbf{a}_k
$m_{k \setminus d}$	Sum over $\mathbf{a}_{k \setminus d}$
$m_{\infty, k}$	Sum over the k th column of \mathbf{A}_∞
$m_{\infty, k \setminus d}$	Sum over $\mathbf{a}_{\infty k \setminus d}$
$m_{s=0, k}$	Number of elements in the k th row of \mathbf{S} that are equal to zero
$m_{s \neq 0, k}$	Number of elements in the k th row of \mathbf{S} that are unequal to zero
N_1	Number of pixels in crosstrack direction
N_2	Number of pixels in track direction
N_C	Number of classes
N_{PT}	Number of temperature levels for parallel tempering
N_t	Number of indicator samples
N_S	Number of samples drawn from the target distribution
N_T	Number of training samples
N_u	Number of actions
N_z	Number of observations
$p(x)$	Probability density function of variable x
P_+	Parameter for $r_{IBP, \text{aug}}$
P_M	Probability of accepting the proposal in Metropolis sampling

$P_{\text{PT}}^{(ij)}$	Probability of accepting a swap of the i th and j th chain
$q(\boldsymbol{\theta} \boldsymbol{\theta}')$	Proposal distribution for $\boldsymbol{\theta}$ given $\boldsymbol{\theta}'$
r_{β_a}	Acceptance ratio for accepting β_a'
r_{IBP}	Acceptance ratio for the IBP
$r_{\text{IBP, aug}}$	Augmented acceptance ratio
r_{M}	Acceptance ratio of the Metropolis sampler
r_{MH}	Acceptance ratio of the Metropolis-Hastings sampler
$r_{\text{PT}}^{(ij)}$	Acceptance ratio for the Metropolis step in parallel tempering
R	Reward
$R_{\text{G, lat}}$	Lateral resolution of the occupancy grid
$R_{\text{G, long}}$	Longitudinal resolution of the occupancy grid
s	Abundance / feature coefficient / sparse coefficient / substate
$s_{k,n}^+$	Proposed abundance / substate
\check{s}	Element of the substate space
\mathbf{s}	Abundance / feature coefficient / sparse coefficient / substate vector
\mathbf{S}	Abundance / feature coefficient / sparse coefficient / substate matrix
\mathbf{S}^+	Abundances / substate matrix of newly proposed features
$\hat{\mathbf{S}}$	Estimate of the abundance / feature coefficient / sparse coefficient / substate matrix
SID	Spectral information divergence
$\overline{\text{SID}}$	Average spectral information divergence
t	Indicator of a feature policy
t_{F}	Time stamp
T_{Corr}	Threshold on the correlation of the endmembers
T_{PT}	Temperature in parallel tempering
u	Action
u^*	Action to infer
\mathbf{u}	Vector of actions
$v_{\text{lat, min}}$	Minimum resolvable lateral speed

$v_{\text{long,min}}$	Minimum resolvable longitudinal speed
v_{H}	Speed of the host vehicle
v_{O}	Speed of the other user's vehicle
$V_{\mathcal{R}}$	Volume of \mathcal{R}
$\mathbf{v}_d^{(k)}$	Eigenvector (of $\Sigma^{(k)}$)
\mathbf{V}	Eigenvectors
$\mathbf{V}^{(k)}$	Eigenvectors of the k th cluster
$\tilde{\mathbf{V}}^{(k)}$	Low-dimensional eigenvectors of the k th cluster
\mathbf{V}_{SC}	Transformed observations in SC
\mathbf{w}	Feature weight vector
\mathbf{w}^+	Proposed feature weight vector
\mathbf{w}_{SVM}	Vector spanning the hyperplane of an SVM
$\hat{\mathbf{w}}_{\text{SVM}}$	Learned \mathbf{w}_{SVM}
\mathbf{W}	Endmember/feature weight matrix
\mathbf{W}^+	Proposed endmember/feature weight matrix
x	Low-dimensional sparse coefficient
\tilde{x}	Low-dimensional sparse coefficient (reduced)
\mathbf{x}	Low-dimensional sparse coefficient vector
$\hat{\mathbf{x}}$	Estimate of \mathbf{x}
$\tilde{\mathbf{x}}$	Low-dimensional sparse coefficient (reduced)
\mathbf{X}	Sparse coefficient matrix / observation matrix (noise-free)
$\hat{\mathbf{X}}$	Estimated sparse coefficient matrix
y	Class label / regression value
y^*	Latent class label
y_{T}	Class label which is contained in \mathcal{D}_{T}
z	Element of an observation vector
Z_{θ}	Normalization constant for variable θ
Z_{IBP}	Normalization for feature classes
\mathbf{z}	Observed (signal) vector
\mathbf{z}^*	New observation vector
$\hat{\mathbf{z}}$	Estimate of \mathbf{z}
\mathbf{z}_{T}	Training vector

\mathbf{Z}	Observation matrix
\mathbf{Z}_T	Observation matrix (training data)
\mathbf{Z}_{HSI}	Observed hyperspectral image

Functions and Operators

$-$	Placeholder for the set of conditional variables
$\binom{n}{k}$	Binomial coefficient
\odot	Hadamard (element-wise) product
$x!$	Faculty of x
$\ \cdot\ _1$	l_1 -norm of a vector
$\ \cdot\ _2$	l_2 -norm of a vector
$\ \cdot\ _F$	Frobenius norm
$[\cdot]^+$	Selector for positive terms
$[\cdot]^-$	Selector for negative terms
$[\cdot]^K$	Selects the K top rows of the argument (matrix) and sets the other rows to zero
∂	Partial derivative
∇	Gradient operator
ϕ_{SC}	Transform into a low-dimensional space
$\mathbf{1}(\cdot, \cdot)$	Indicator function
$H(\cdot)$	Heaviside step function or unit step function
$\arccos(\cdot)$	Inverse cosine function
$B(\cdot, \cdot)$	Beta function
$\text{ch}(\cdot)$	Set of children variables of a variable in a Bayesian network
$\text{diag}(\cdot)$	Diagonal matrix from a vector
$E_{\text{Emp}}(f)$	Empirical risk over f
$E_{\text{Orth}}(\cdot)$	Orthogonality penalty term
$E_{\text{Risk}}(f)$	Expected risk over f
$E_{\text{Sparse}}(\cdot)$	Sparsity penalty term
$f_{\boldsymbol{\theta}}(\cdot)$	Prediction function with parameter $\boldsymbol{\theta}$
$f_{\text{SVM}}(\cdot)$	Linear discriminant function (SVM)
$K_{\text{SVM}}(\cdot)$	Kernel function (SVM)
$K_{\text{RBF}}(\cdot)$	Radial basis function kernel

$\mathbb{L}(\cdot)$	Loss function
$L(\cdot)$	Lagrangian (SVM)
$L_{\text{Dual}}(\cdot)$	Lagrangian in <i>Wolfe dual representation</i> (SVM)
$\text{median}(\cdot)$	Median
$\min(a, b)$	Algorithmic notation for the minimum of a and b
$\text{normalize}()$	Algorithmic notation for normalization (each column sums to one)
$\text{pa}(\cdot)$	Set of parent variables of a variable in a Bayesian network
$U(\cdot)$	Energy term of the target pdf (augmented)
$V(\cdot)$	Energy term of the target pdf (not augmented)

Probability Density and Mass Functions

$\text{Bin}_m(\theta)$	Binomial distribution for m with parameter θ
$\text{Beta}_{\boldsymbol{\theta}}(\alpha, N)$	Beta distribution for $\boldsymbol{\theta}$ with parameters α and N
$\text{BetaBin}_m(\alpha, N)$	Beta-Binomial distribution for m with parameters α and N
$\text{Cat}_s(\boldsymbol{\theta})$	Categorical distribution for s with parameters $\boldsymbol{\theta}$
$\text{Dir}_{\boldsymbol{\theta}}(\boldsymbol{\alpha})$	Dirichlet distribution for $\boldsymbol{\theta}$ with parameters $\boldsymbol{\alpha}$
$\text{DirMult}_{\mathbf{m}}(\boldsymbol{\alpha})$	Dirichlet-Multinomial distribution for m with parameters $\boldsymbol{\alpha}$
$\text{Exp}_w(\lambda)$ ¹	Exponential distribution for w with parameter λ
$\text{Ga}_{\tau}(\alpha, \beta)$ ²	Gamma distribution for τ with parameters α and β
$\mathcal{N}_x(\mu, \sigma^2)$	Gaussian distribution for x with mean μ and variance σ^2
$\text{IGa}_{\sigma}(\alpha, \beta)$ ³	Inverse Gamma distribution for σ with parameters α and β
$\text{Laplace}_x(a, b)$	Laplace distribution for x with parameters a and b
$\text{Poisson}_K(\alpha)$	Poisson distribution for K with parameter α
$\mathcal{TN}_x(\mu, \sigma)$	Truncated Gaussian distribution for x with mean μ and variance σ^2 with $\mathbf{x} \in \mathbb{R}_+$
$\mathcal{U}_x(a, b)$	Uniform distribution for x in the range from a to b

¹ $\text{Exp}_w(\lambda) = \frac{1}{\lambda} \exp\{-\frac{1}{\lambda}w\}$

² $\text{Ga}_{\tau}(\alpha, \beta) = \frac{\beta^{\alpha}}{\Gamma(\alpha)} \tau^{\alpha-1} \exp\{-\beta\tau\}$

³ $\text{IGa}_{\sigma}(\alpha, \beta) = \frac{\beta^{\alpha}}{\Gamma(\alpha)} \sigma^{-\alpha-1} \exp\{-\frac{\beta}{\sigma}\}$

Bibliography

- [1] R. A. Schowengerdt, *Remote Sensing: Models and Methods for Image Processing*, Elsevier, 3rd edition, 2007.
- [2] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan, “Learning influence probabilities in social networks,” in *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*, New York City, NY, USA, Feb. 2010, pp. 241–250.
- [3] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, “Evaluating collaborative filtering recommender systems,” *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 5–53, 2004.
- [4] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, “A survey of robot learning from demonstration,” *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.
- [5] G. Hughes, “On the mean accuracy of statistical pattern recognizers,” *IEEE Transactions on Information Theory*, vol. 14, no. 1, pp. 55–63, Jan 1968.
- [6] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.
- [7] J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, “Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 5, no. 2, pp. 354–379, 2012.
- [8] J. Hahn and A. M. Zoubir, “Bayesian nonparametric unmixing of hyperspectral images,” *IEEE Transactions on Geoscience and Remote Sensing (under review)*, 2016.
- [9] K. Pearson, “On lines and planes of closest fit to systems of points in spaces,” *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science, Sixth Series*, vol. 2, pp. 559–572, 1901.
- [10] I. T. Jolliffe, *Principal component analysis*, Springer, New York, 2nd edition, 2002.
- [11] A. Hyvärinen and E. Oja, “Independent component analysis: algorithms and applications,” *Neural networks*, vol. 13, no. 4, pp. 411–430, 2000.
- [12] D. D. Lee and H. S. Seung, “Learning the parts of objects by non-negative matrix factorization,” *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.
- [13] D. M. Blei, P. R. Cook, and M. Hoffman, “Bayesian nonparametric matrix factorization for recorded music,” in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, June 2010, pp. 439–446.

- [14] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.
- [15] Z. Ghahramani and T. L. Griffiths, “Infinite latent feature models and the Indian buffet process,” in *Advances in Neural Information Processing Systems 18*, 2005, pp. 475–482.
- [16] K. Miller, M. I. Jordan, and T. L. Griffiths, “Nonparametric latent feature models for link prediction,” in *Advances in Neural Information Processing Systems 22*, 2009, pp. 1276–1284.
- [17] M. N. Schmidt, O. Winther, and L. K. Hansen, “Bayesian non-negative matrix factorization,” in *Proceedings of the 8th International Conference on Independent Component Analysis and Signal Separation*, Paraty, Brazil, Mar. 2009, pp. 540–547.
- [18] B. A. Olshausen and D. J. Field, “Sparse coding with an overcomplete basis set: A strategy employed by v1?,” *Vision research*, vol. 37, no. 23, pp. 3311–3325, 1997.
- [19] H. Lee, A. Battle, R. Raina, and A. Y. Ng, “Efficient sparse coding algorithms,” in *Advances in Neural Information Processing Systems 19*, 2006, pp. 801–808.
- [20] J. Yang, K. Yu, Y. Gong, and T. Huang, “Linear spatial pyramid matching using sparse coding for image classification,” in *Proceedings of the 22nd IEEE International Conference on Computer Vision and Pattern Recognition*, Miami Beach, FL, USA, June 2009, pp. 1794–1801.
- [21] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan, “Sparse representation for computer vision and pattern recognition,” *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1031–1044, 2010.
- [22] E. C. Smith and M. S. Lewicki, “Efficient auditory coding,” *Nature*, vol. 439, no. 7079, pp. 978–982, 2006.
- [23] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, “Locality-constrained linear coding for image classification,” in *Proceedings of the 23rd IEEE International Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, USA, June 2010, pp. 3360–3367.
- [24] T. J. Mitchell and J. J. Beauchamp, “Bayesian variable selection in linear regression,” *Journal of the American Statistical Association*, vol. 83, no. 404, pp. 1023–1032, 1988.
- [25] H. Ishwaran and J. S. Rao, “Spike and slab variable selection: Frequentist and Bayesian strategies,” *Annals of Statistics*, pp. 730–773, 2005.
- [26] I. Goodfellow, A. Courville, and Y. Bengio, “Large-scale feature learning with spike-and-slab sparse coding,” in *Proceedings of the 29th International Conference on Machine Learning*, New York, NY, USA, July 2012, pp. 1439–1446.

- [27] I. J. Goodfellow, A. Courville, and Y. Bengio, “Scaling up spike-and-slab models for unsupervised feature learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1902–1914, 2013.
- [28] P. J. Werbos, “Applications of advances in nonlinear sensitivity analysis,” in *Proceedings of the 10th IFIP Conference on System Modeling and Optimization*, New York, NY, USA, Sept. 1981, pp. 762–770.
- [29] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.
- [30] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, “Why does unsupervised pre-training help deep learning?,” *Journal of Machine Learning Research*, vol. 11, pp. 625–660, 2010.
- [31] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
- [32] Y. Bengio, “Learning deep architectures for AI,” *Foundations and Trends[®] in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.
- [33] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, “Learning hierarchical features for scene labeling,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1915–1929, 2013.
- [34] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the 27th IEEE International Conference on Computer Vision and Pattern Recognition*, Columbus, OH, June 2014.
- [35] Q. Le, M. Ranzato, R. Monga, M. Devin, G. Corrado, K. Chen, J. Dean, and A. Ng, “Building high-level features using large scale unsupervised learning,” in *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland, June 2012.
- [36] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.
- [37] A. Mohamed, G. E. Dahl, and G. Hinton, “Acoustic modeling using deep belief networks,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14–22, 2012.
- [38] G. E. Dahl, D. Yu, L. Deng, and A. Acero, “Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.
- [39] L. Wiskott and T. J. Sejnowski, “Slow feature analysis: Unsupervised learning of invariances,” *Neural Computation*, vol. 14, no. 4, pp. 715–770, 2002.

- [40] N. Sprague, “Contingent features for reinforcement learning,” in *Proceedings of the 24th International Conference on Artificial Neural Networks and Machine Learning*, Hamburg, Germany, Sept. 2014.
- [41] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng, “Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis,” in *Proceedings of the 24th IEEE International Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, USA, June 2011.
- [42] M. A. Casey and A. Westner, “Separation of mixed audio sources by independent subspace analysis,” in *Proceedings of the International Computer Music Conference*, 2000, pp. 154–161.
- [43] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [44] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, , N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, “Mastering the game of go with deep neural networks and tree search,” *Nature*, vol. 529, pp. 484–489, 2016.
- [45] E. J. Candès, J. K. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements,” *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.
- [46] D. L. Donoho, “Compressed sensing,” *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.
- [47] J. M. Duarte-Carvajalino and G. Sapiro, “Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization,” *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1395–1408, 2009.
- [48] S. Gleichman and Y. C. Eldar, “Blind compressed sensing,” *IEEE Transactions on Information Theory*, vol. 57, no. 10, pp. 6958–6975, Oct. 2011.
- [49] V. N. Vapnik, “An overview of statistical learning theory,” *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988–999, 1999.
- [50] O. Bousquet, S. Boucheron, and G. Lugosi, “Introduction to statistical learning theory,” in *Advanced Lectures on Machine Learning*, pp. 169–207. Springer, 2004.
- [51] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [52] L. Wasserman, *All of statistics: a concise course in statistical inference*, Springer Science & Business Media, 2013.

- [53] A. Gelman, J. B. Carlin, H. S. Stern, A. Vehtari, and D. B. Rubin, *Bayesian data analysis*, Chapman & Hall/CRC, 3rd edition, 2013.
- [54] C. M. Bishop, *Pattern recognition and machine learning*, Springer-Verlag New York, 2006.
- [55] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, “Stochastic variational inference,” *Journal of Machine Learning Research*, vol. 14, pp. 1303–1347, 2013.
- [56] W. R. Gilks, S. Richardson, and D. Spiegelhalter, *Markov Chain Monte Carlo*, Chapman & Hall/CRC, 1995.
- [57] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, “Equation of state calculations by fast computing machines,” *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.
- [58] W. K. Hastings, “Monte Carlo sampling methods using Markov chains and their applications,” *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.
- [59] G. O. Roberts, A. Gelman, and W. R. Gilks, “Weak convergence and optimal scaling of random walk Metropolis algorithms,” *Annals of Applied Probability*, vol. 7, no. 1, pp. 110–120, 1997.
- [60] D. Barber, *Bayesian reasoning and machine learning*, Cambridge University Press, 2012.
- [61] S. Geman and D. Geman, “Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 6, pp. 721–741, 1984.
- [62] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.
- [63] R. M. Neal, “Slice sampling,” *Annals of Statistics*, pp. 705–741, 2003.
- [64] R. M. Neal, “MCMC using Hamiltonian dynamics,” *Handbook of Markov Chain Monte Carlo*, vol. 2, pp. 113–162, 2011.
- [65] R. H. Swendsen and J.-S. Wang, “Replica Monte Carlo simulation of spin-glasses,” *Physical Review Letters*, vol. 57, pp. 2607–2609, Nov. 1986.
- [66] D. J. Earl and M. W. Deem, “Parallel tempering: Theory, applications, and new perspectives,” *Physical Chemistry Chemical Physics*, vol. 7, no. 23, pp. 3910–3916, 2005.
- [67] U. Paquet, *Bayesian Inference for Latent Variable Models*, Ph.D. thesis, Wolfson College, University of Cambridge, 2007.
- [68] A. J. Jerri, “The Shannon sampling theorem – its various extensions and applications: A tutorial review,” *Proceedings of the IEEE*, vol. 65, no. 11, pp. 1565–1596, 1977.

- [69] E. J. Candès and T. Tao, “Decoding by linear programming,” *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.
- [70] E. J. Candès, J. Romberg, and T. Tao, “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.
- [71] R. G. Baraniuk, “Compressive sensing [lecture notes],” *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 118–121, 2007.
- [72] B. K. Natarajan, “Sparse approximate solutions to linear systems,” *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, 1995.
- [73] E. J. Candès and J. K. Romberg, “ l_1 -Magic: Recovery of sparse signals,” <http://www.acm.caltech.edu/l1magic/>, 2005.
- [74] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo, “Sparse reconstruction by separable approximation,” *IEEE Transactions on Signal Processing*, vol. 57, no. 7, pp. 2479–2493, 2009.
- [75] D. Knowles and Z. Ghahramani, “Nonparametric Bayesian sparse factor models with application to gene expression modeling,” *Annals of Applied Statistics*, vol. 5, no. 2B, pp. 1534–1552, June 2011.
- [76] S. Williamson, C. Wang, K. A. Heller, and D. M. Blei, “The IBP compound Dirichlet process and its application to focused topic modeling,” in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, June 2010, pp. 1151–1158.
- [77] S. J. Gershman and D. M. Blei, “A tutorial on Bayesian nonparametric models,” *Journal of Mathematical Psychology*, vol. 56, no. 1, pp. 1–12, 2012.
- [78] Z. Ghahramani, P. Sollich, and T. L. Griffiths, “Bayesian nonparametric latent feature models,” in *Bayesian Statistics 8*. 2007, Oxford University Press, Oxford.
- [79] T. L. Griffiths and Z. Ghahramani, “The Indian buffet process: An introduction and review,” *Journal of Machine Learning Research*, vol. 12, pp. 1185–1224, 2011.
- [80] E. W. Forgy, “Cluster analysis of multivariate data: efficiency versus interpretability of classifications,” *Biometrics*, vol. 21, pp. 768–769, 1965.
- [81] S. Lloyd, “Least squares quantization in PCM,” *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, Mar. 1982.
- [82] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, vol. 39, no. 1, pp. 1–38, 1977.
- [83] Y. Boykov, O. Veksler, and R. Zabih, “Fast approximate energy minimization via graph cuts,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

- [84] L. R. Ford and D. R. Fulkerson, “Maximal flow through a network,” *Canadian Journal of Mathematics*, vol. 8, no. 3, pp. 399–404, 1956.
- [85] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.
- [86] U. Luxburg, “A tutorial on spectral clustering,” *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.
- [87] P. Perona and L. Zelnik-Manor, “Self-tuning spectral clustering,” in *Advances in Neural Information Processing Systems 17*, 2004, vol. 17, pp. 1601–1608.
- [88] A. Y. Ng, M. I. Jordan, and Y. Weiss, “On spectral clustering: Analysis and an algorithm,” in *Advances in Neural Information Processing Systems 15*, 2002, pp. 849–856.
- [89] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1996.
- [90] V. K. Goyal and M. Vetterli, “Block transform adaptation by stochastic gradient descent,” in *Proceedings of the IEEE Digital Signal Processing Workshop*, 1998.
- [91] T. Cover and P. Hart, “Nearest neighbor pattern classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 13, no. 1, pp. 21–27, 1967.
- [92] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, Pittsburgh, PA, USA, July 1992, pp. 144–152.
- [93] P. Wolfe, “A duality theorem for non-linear programming,” *Quarterly of Applied Mathematics*, pp. 239–244, 1961.
- [94] J. Shawe-Taylor and N. Cristianini, *Kernel methods for pattern analysis*, Cambridge University Press, 2004.
- [95] M. Stone, “Cross-validatory choice and assessment of statistical predictions,” *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, pp. 111–147, 1974.
- [96] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, “A Practical Guide to Support Vector Classification,” 2000.
- [97] V. N. Vapnik, *Statistical Learning Theory*, Wiley-Interscience, 1998.
- [98] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, Cambridge, MA, USA, 2001.
- [99] C.-W. Hsu and C.-J. Lin, “A comparison of methods for multiclass support vector machines,” *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

- [100] V. Kumar, J. Hahn, and A.M. Zoubir, “Band selection for hyperspectral images based on self-tuning spectral clustering,” in *Proceedings of the 21st European Signal Processing Conference*, Marrakech, Morocco, Sept. 2013.
- [101] J. Hahn, S. Rosenkranz, and A. M. Zoubir, “Adaptive compressed classification for hyperspectral imagery,” in *Proceedings of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing*, Florence, Italy, May 2014, pp. 1020–1024.
- [102] G. Shaw and D. Manolakis, “Signal processing for hyperspectral image exploitation,” *IEEE Signal Processing Magazine*, vol. 19, no. 1, pp. 12–16, 2002.
- [103] G. Elmasry, M. Kamruzzaman, D. W. Sun, and P. Allen, “Principles and applications of hyperspectral imaging in quality evaluation of agro-food products: a review,” *Critical Reviews in Food Science and Nutrition*, vol. 52, no. 11, pp. 999–1023, 2012.
- [104] S. Serranti, A. Gargiulo, and G. Bonifazi, “Characterization of post-consumer polyolefin wastes by hyperspectral imaging for quality control in recycling processes,” *Waste Management*, vol. 31, no. 11, pp. 2217 – 2227, 2011.
- [105] Y. Tarabalka, J. Chanussot, and J. A. Benediktsson, “Segmentation and classification of hyperspectral images using watershed transformation,” *Pattern Recognition*, vol. 43, no. 7, pp. 2367 – 2379, 2010.
- [106] M. Fauvel, J. A. Benediktsson, J. Chanussot, and J. R. Sveinsson, “Spectral and spatial classification of hyperspectral data using SVMs and morphological profiles,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 46, no. 11, pp. 3804–3814, 2008.
- [107] M. Fauvel, Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton, “Advances in spectral-spatial classification of hyperspectral images,” *Proceedings of the IEEE*, vol. 101, no. 3, pp. 652–675, 2013.
- [108] C. Debes, A. Merentitis, R. Heremans, J. Hahn, N. Frangiadakis, T. van Kasteren, W. Liao, R. Bellens, A. Piurica, S. Gautama, W. Philips, S. Prasad, Q. Du, and F. Pacifici, “Hyperspectral and LiDAR data fusion: Outcome of the 2013 GRSS data fusion contest,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 2405–2418, June 2014.
- [109] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, “Deep learning-based classification of hyperspectral data,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 2094–2107, June 2014.
- [110] Q. Zou, L. Ni, T. Zhang, and Q. Wang, “Deep learning based feature selection for remote sensing scene classification,” *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 11, pp. 2321–2325, Nov. 2015.
- [111] A. Romero, C. Gatta, and G. Camps-Valls, “Unsupervised deep feature extraction for remote sensing image classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 3, pp. 1349–1362, Mar. 2016.

- [112] L. Zhang, L. Zhang, and B. Du, “Deep learning for remote sensing data: A technical tutorial on the state of the art,” *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, no. 2, pp. 22–40, June 2016.
- [113] P. Bajcsy and P. Groves, “Methodology for hyperspectral band selection,” *Sensing Journal*, vol. 70, pp. 793–802, 2004.
- [114] C.-I Chang, Q. Du, T.-L. Sun, and M. L. G. Althouse, “A joint band prioritization and band-decorrelation approach to band selection for hyperspectral image classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 37, no. 6, pp. 2631–2641, 1999.
- [115] C.-I Chang and S. Wang, “Constrained band selection for hyperspectral imagery,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 44, no. 6, pp. 1575–1585, 2006.
- [116] C.-I Chang and Q. Du, “Estimation of number of spectrally distinct signal sources in hyperspectral imagery,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, no. 3, pp. 608–619, 2004.
- [117] A. Martinez-Uso, F. Pla, J. M. Sotoca, and P. Garcia-Sevilla, “Clustering-based hyperspectral band selection using information measures,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 12, pp. 4158–4171, 2007.
- [118] Y. Qian, F. Yao, and S. Jia, “Band selection for hyperspectral imagery using affinity propagation,” *IET Computer Vision*, vol. 3, no. 4, pp. 213–222, 2009.
- [119] M. Golbabaee, S. Arberet, and P. Vandergheynst, “Multichannel compressed sensing via source separation for hyperspectral images,” in *Proceedings of the 18th European Signal Processing Conference*, Aalborg, Denmark, Aug. 2010.
- [120] C. Li, T. Sun, K. F. Kelly, and Y. Zhang, “A compressive sensing and unmixing scheme for hyperspectral data processing,” *IEEE Transactions on Image Processing*, vol. 21, no. 3, pp. 1200–1210, 2012.
- [121] J. Hahn, C. Debes, M. Leigsnering, and A. M. Zoubir, “Compressive sensing and adaptive direct sampling in hyperspectral imaging,” *Digital Signal Processing*, vol. 26, pp. 113–126, 2014.
- [122] M. A. Davenport, M. F. Duarte, M. Wakin, J. Laska, D. Takhar, K. Kelly, and R. Baraniuk, “The smashed filter for compressive classification and target recognition,” in *Proceedings of the SPIE 6498, Computational Imaging V*, 2007, vol. 6498.
- [123] “AVIRIS NW Indiana’s Indian Pines 1992 data set [Online], Available: <ftp://ftp.ecn.purdue.edu/biehl/MultiSpec/92AV3C> (original files) and <ftp://ftp.ecn.purdue.edu/biehl/PCMultiSpec/ThyFiles.zip> (ground truth),” .
- [124] S. Tadjudin and D. A. Landgrebe, “Covariance estimation with limited training samples,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 37, no. 4, pp. 2113–2118, 1999.

- [125] P. Gamba, “A collection of data for urban area characterization,” in *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium*, Anchorage, AK, USA, Sept. 2004, vol. 1.
- [126] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 3rd edition, 2006.
- [127] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [128] E. J. Candès and M. B. Wakin, “An introduction to compressive sampling,” *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, 2008.
- [129] R. Calderbank, S. Jafarpour, and R. Schapire, “Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain,” Tech. Rep., Computer Science, Princeton University, 2009.
- [130] G. A. Shaw and H. K. Burke, “Spectral imaging for remote sensing,” *Lincoln Laboratory Journal*, vol. 14, no. 1, pp. 3–28, 2003.
- [131] D. Takhar, J. N. Laska, M. B. Wakin, M. F. Duarte, D. Baron, S. Sarvotham, K. F. Kelly, and R. G. Baraniuk, “A new compressive imaging camera architecture using optical-domain compression,” in *Proceedings of the SPIE 6065, Computational Imaging IV*. International Society for Optics and Photonics, 2006, pp. 606509–606509.
- [132] D. D. Lee and H. S. Seung, “Algorithms for non-negative matrix factorization,” in *Advances in Neural Information Processing Systems 14*, 2001, pp. 556–562.
- [133] C. Eckart and G. Young, “The approximation of one matrix by another of lower rank,” *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.
- [134] M. Arngren, M. N. Schmidt, and J. Larsen, “Unmixing of hyperspectral images using Bayesian non-negative matrix factorization with volume prior,” *Journal of Signal Processing Systems*, vol. 65, no. 3, pp. 479–496, 2010.
- [135] N. Dobigeon, S. Moussaoui, M. Coulon, J.-Y. Tournier, and A. O. Hero, “Joint Bayesian endmember extraction and linear unmixing for hyperspectral imagery,” *IEEE Transactions on Signal Processing*, vol. 57, no. 11, pp. 4355–4368, Nov. 2009.
- [136] J. W. Boardman, “Automating spectral unmixing of AVIRIS data using convex geometry concepts,” in *Proceedings of the 4th JPL Airborne Geoscience Workshop*, Washington, DC, United States, Oct. 1993.
- [137] M. E. Winter, “N-FINDR: an algorithm for fast autonomous spectral end-member determination in hyperspectral data,” in *Proceedings of the SPIE 3753, Imaging Spectrometry V*. International Society for Optics and Photonics, 1999, pp. 266–275.

- [138] J. M. P. Nascimento and J. M. Bioucas-Dias, "Vertex component analysis: A fast algorithm to unmix hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 4, pp. 898–910, 2005.
- [139] M. Berman, H. Kiiveri, R. Lagerstrom, A. Ernst, R. Dunne, and J.F. Huntington, "ICE: a statistical approach to identifying endmembers in hyperspectral images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, no. 10, pp. 2085–2095, Oct. 2004.
- [140] M. D. Craig, "Minimum-volume transforms for remotely sensed data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 32, no. 3, pp. 542–552, 1994.
- [141] T. H. Chan, C. Y. Chi, Y. M. Huang, and W. K. Ma, "A convex analysis-based minimum-volume enclosing simplex algorithm for hyperspectral unmixing," *IEEE Transactions on Signal Processing*, vol. 57, no. 11, pp. 4418–4432, Nov. 2009.
- [142] V. P. Pauca, J. Piper, and R. J. Plemmons, "Nonnegative matrix factorization for spectral data analysis," *Linear Algebra and its Applications*, vol. 416, no. 1, pp. 29–47, 2006.
- [143] R. B. Singer and T. B. McCord, "Mars-large scale mixing of bright and dark surface materials and implications for analysis of spectral reflectance," in *Lunar and Planetary Science Conference Proceedings*, Houston, TX, USA, Mar. 1979, vol. 10, pp. 1835–1848.
- [144] A. Halimi, Y. Altmann, N. Dobigeon, and J.-Y. Tourneret, "Nonlinear unmixing of hyperspectral images using a generalized bilinear model," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 11, pp. 4153–4162, 2011.
- [145] Y. Altmann, N. Dobigeon, S. McLaughlin, and J.-Y. Tourneret, "Nonlinear spectral unmixing of hyperspectral images using Gaussian processes," *IEEE Transactions on Signal Processing*, vol. 61, no. 10, pp. 2442–2453, 2013.
- [146] N. Dobigeon, J.-Y. Tourneret, C. Richard, J. C. M. Bermudez, S. McLaughlin, and A. O. Hero, "Nonlinear unmixing of hyperspectral images: Models and algorithms," *IEEE Signal Processing Magazine*, vol. 31, no. 1, pp. 82–94, Jan. 2014.
- [147] Y. Altmann, M. Pereyra, and S. McLaughlin, "Bayesian nonlinear hyperspectral unmixing with spatial residual component analysis," *IEEE Transactions on Computational Imaging*, vol. 1, no. 3, pp. 174–185, 2015.
- [148] B. Hapke, *Theory of reflectance and emittance spectroscopy*, Cambridge University Press, 2012.
- [149] N. Dobigeon, J.-Y. Tourneret, and C.-I. Chang, "Semi-supervised linear spectral unmixing using a hierarchical Bayesian model for hyperspectral imagery," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 2684–2695, July 2008.

- [150] K. E. Themelis, A. A. Rontogiannis, and K. D. Koutroumbas, “A novel hierarchical Bayesian approach for sparse semisupervised hyperspectral unmixing,” *IEEE Transactions on Signal Processing*, vol. 60, no. 2, pp. 585–599, Feb. 2012.
- [151] H. Akaike, “A new look at the statistical model identification,” *IEEE Transactions on Automatic Control*, vol. 19, no. 6, pp. 716–723, 1974.
- [152] G. Schwarz, “Estimating the dimension of a model,” *Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.
- [153] J. M. Bioucas-Dias and J. M. P. Nascimento, “Hyperspectral subspace identification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 46, no. 8, pp. 2435–2445, 2008.
- [154] B. Luo, J. Chanussot, S. Douté, and L. Zhang, “Empirical automatic estimation of the number of endmembers in hyperspectral images,” *IEEE Geoscience and Remote Sensing Letters*, vol. 10, no. 1, pp. 24–28, 2013.
- [155] R. Heylen and P. Scheunders, “Hyperspectral intrinsic dimensionality estimation with nearest-neighbor distance ratios,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 6, no. 2, pp. 570–579, Apr. 2013.
- [156] A. Zare and P. Gader, “Sparsity promoting iterated constrained endmember detection in hyperspectral imagery,” *IEEE Geoscience and Remote Sensing Letters*, vol. 4, no. 3, pp. 446, 2007.
- [157] C.-I. Chang, “Further results on relationship between spectral unmixing and subspace projection,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 36, no. 3, pp. 1030–1032, 1998.
- [158] N. Chopin, “Fast simulation of truncated Gaussian distributions,” *Statistics and Computing*, vol. 21, no. 2, pp. 275–288, 2010.
- [159] E. Meeds, Z. Ghahramani, R. M. Neal, and S. T. Roweis, “Modeling dyadic data with binary latent factors,” in *Advances in Neural Information Processing Systems 19*, 2006, pp. 977–984.
- [160] F. Doshi-Velez and Z. Ghahramani, “Accelerated sampling for the Indian buffet process,” in *Proceedings of the 26th Annual International Conference on Machine Learning*, New York, NY, USA, June 2009, pp. 273–280.
- [161] L. Devroye, *Non-Uniform Random Variate Generation*, Springer-Verlag, New York, 1986.
- [162] P. J. Green, “Reversible jump Markov chain Monte Carlo computation and Bayesian model determination,” *Biometrika*, vol. 82, no. 4, pp. 711–732, 1995.
- [163] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, “Optimization by simulated annealing,” *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

- [164] V. Černý, “Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm,” *Journal of Optimization Theory and Applications*, vol. 45, no. 1, pp. 41–51, 1985.
- [165] R. N. Clark, G. A. Swayze, R. Wise, E. Livo, T. Hoefen, R. Kokaly, and S. J. Sutley, “USGS digital spectral library splib06a: U.S. Geological Survey, Digital Data Series 231, <http://speclab.cr.usgs.gov/spectral.lib06.>,” Tech. Rep., 2007.
- [166] C.-I Chang, “An information-theoretic approach to spectral variability, similarity, and discrimination for hyperspectral image analysis,” *IEEE Transactions on Information Theory*, vol. 46, no. 5, pp. 1927–1932, Aug. 2000.
- [167] S. Kullback, *Information theory and statistics*, John Wiley and Sons, Inc., New York, NY, USA, 1968.
- [168] A. M. Baldridge, S. J. Hook, C. I. Grove, and G. Rivera, “The ASTER spectral library version 2.0,” *Remote Sensing of Environment*, vol. 113, no. 4, pp. 711 – 715, 2009.
- [169] E. Christophe, D. Leger, and C. Mailhes, “Quality criteria benchmark for hyperspectral imagery,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 9, pp. 2103–2114, Sept. 2005.
- [170] T. Akgun, Y. Altunbasak, and R. M. Mersereau, “Super-resolution reconstruction of hyperspectral images,” *IEEE Transactions on Image Processing*, vol. 14, no. 11, pp. 1860–1875, Nov. 2005.
- [171] R. Thibaux and M. I. Jordan, “Hierarchical Beta processes and the Indian buffet process,” in *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, San Juan, Puerto Rico, Mar. 2007, pp. 564–571.
- [172] L. Carin, D. M. Blei, and J. W. Paisley, “Variational inference for stick-breaking Beta process priors,” in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, June 2011, pp. 889–896.
- [173] R. Vilalta and Y. Drissi, “A perspective view and survey of meta-learning,” *Artificial Intelligence Review*, vol. 18, no. 2, pp. 77–95, 2002.
- [174] G. Shani, D. Heckerman, and R. I. Brafman, “An MDP-based recommender system,” *Journal of Machine Learning Research*, vol. 6, pp. 1265–1295, 2005.
- [175] J. Hahn and A. M. Zoubir, “Risk-sensitive decision making via constrained expected returns,” in *Proceedings of the 41st IEEE International Conference on Acoustics, Speech and Signal Processing*, Shanghai, China, Mar. 2016, pp. 2569–2573.
- [176] A. Y. Ng and S. J. Russell, “Algorithms for inverse reinforcement learning,” in *Proceedings of the 17th International Conference on Machine Learning*, San Francisco, CA, USA, June 2000, pp. 663–670.

- [177] J. Hahn and A. M. Zoubir, “Inverse reinforcement learning using expectation maximization in mixture models,” in *Proceedings of the 40th IEEE International Conference on Acoustics, Speech and Signal Processing*, Brisbane, Australia, Apr. 2015, pp. 3721–3725.
- [178] R. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, NJ, USA, 1st edition, 1957.
- [179] D. A. Pomerleau, “Efficient training of artificial neural networks for autonomous navigation,” *Neural Computation*, vol. 3, no. 1, pp. 88–97, 1991.
- [180] S. J. Bradtke and A. G. Barto, “Linear least-squares algorithms for temporal difference learning,” *Machine Learning*, vol. 22, no. 1-3, pp. 33–57, 1996.
- [181] S. J. Gershman, K. A. Norman, and Y. Niv, “Discovering latent causes in reinforcement learning,” *Current Opinion in Behavioral Sciences*, vol. 5, pp. 43 – 50, 2015.
- [182] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, “Planning and acting in partially observable stochastic domains,” *Artificial Intelligence*, vol. 101, no. 1, pp. 99–134, 1998.
- [183] R. D. Smallwood and E. J. Sondik, “The optimal control of partially observable Markov processes over a finite horizon,” *Operations Research*, vol. 21, no. 5, pp. 1071–1088, 1973.
- [184] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*, MIT Press, Cambridge, MA, USA, 1998.
- [185] C. J. C. H. Watkins, *Learning from Delayed Rewards*, Ph.D. thesis, Cambridge University, Cambridge, England., 1989.
- [186] M. Petrik, G. Taylor, R. Parr, and S. Zilberstein, “Feature selection using regularization in approximate linear programs for Markov decision processes,” in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, June 2010, pp. 871–878.
- [187] M. Riedmiller, “Neural fitted Q iteration – first experiences with a data efficient neural reinforcement learning method,” in *Proceedings of the 16th European Conference on Machine Learning*, Porto, Portugal, Oct. 2005, pp. 317–328.
- [188] D.-R. Liu, H.-L. Li, and D. Wang, “Feature selection and feature learning for high-dimensional batch reinforcement learning: A survey,” *International Journal of Automation and Computing*, vol. 12, no. 3, pp. 229–242, 2015.
- [189] M. Hutter, “Feature reinforcement learning: Part I. Unstructured MDPs,” *Journal of Artificial General Intelligence*, vol. 1, no. 1, pp. 3–24, 2009.
- [190] M. Daswani, P. Sunehag, and M. Hutter, “Feature reinforcement learning: State of the art,” in *AAAI-14 Workshop on Sequential Decision-making with Big Data*. Association for the Advancement of Artificial Intelligence, 2014.

- [191] C. Boutilier, R. Dearden, and M. Goldszmidt, “Exploiting structure in policy construction,” in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada, Aug. 1995, pp. 1104–1111.
- [192] T. Degris, O. Sigaud, and P.-H. Wuillemin, “Learning the structure of factored Markov decision processes in reinforcement learning problems,” in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, USA, June 2006, pp. 257–264.
- [193] T. Nguyen, Z. Li, T. Silander, and T. Y. Leong, “Online feature selection for model-based reinforcement learning,” in *Proceedings of the 30th International Conference on Machine Learning*, Atlanta, GA, USA, June 2013, pp. 498–506.
- [194] M. Yuan and Y. Lin, “Model selection and estimation in regression with grouped variables,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.
- [195] M. Wulfmeier, P. Ondruska, and I. Posner, “Deep inverse reinforcement learning,” in *NIPS Deep Reinforcement Learning Workshop*, Montréal, Canada, Dec. 2015.
- [196] J. Choi and K.-E. Kim, “Nonparametric Bayesian inverse reinforcement learning for multiple reward functions,” in *Advances in Neural Information Processing Systems 25*, 2012, pp. 305–313.
- [197] B. Michini and J. P. How, “Bayesian nonparametric inverse reinforcement learning,” in *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, Bristol, UK, Sept. 2012, pp. 148–163.
- [198] A. Surana and K. Srivastava, “Bayesian nonparametric inverse reinforcement learning for switched Markov decision processes,” in *Proceedings of the 13th International Conference on Machine Learning and Applications*, Dec. 2014, pp. 47–54.
- [199] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, “Maximum margin planning,” in *Proceedings of the 23rd International Conference on Machine Learning*, New York, NY, USA, June 2006, pp. 729–736.
- [200] M. Lopes, F. S. Melo, and L. Montesano, “Affordance-based imitation learning in robots,” in *Proceedings of the 20th IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, USA, Oct. 2007, pp. 1015–1021.
- [201] S. Ross, G. J. Gordon, and D. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” in *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, Ft. Lauderdale, FL, USA, Apr. 2011, pp. 627–635.
- [202] N. Ratliff, D. Bradley, J. A. Bagnell, and J. Chestnutt, “Boosting structured prediction for imitation learning,” in *Advances in Neural Information Processing Systems 20*, 2007.

- [203] S. Parisi, M. Pirotta, N. Smacchia, L. Bascetta, and M. Restelli, “Policy gradient approaches for multi-objective sequential decision making,” in *Proceedings of the International Joint Conference on Neural Networks*, Beijing, China, July 2014, pp. 2323–2330.
- [204] P. Vamplew, R. Dazeley, E. Barker, and A. Kelarev, “Constructing stochastic mixture policies for episodic multiobjective reinforcement learning tasks,” in *Proceedings of the 22nd Australasian Joint Conference on Advances in Artificial Intelligence*, Melbourne, Australia, Dec. 2009, pp. 340–349.
- [205] C. R. Shelton, *Importance sampling for reinforcement learning with multiple objectives*, Ph.D. thesis, Massachusetts Institute of Technology, 2001.
- [206] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.
- [207] A. Šošić, A. M. Zoubir, and H. Koepl, “A Bayesian approach to policy recognition and state representation learning,” *arXiv:1605.01278*, 2016.
- [208] G. McLachlan and D. Peel, *Finite Mixture Models*, John Wiley & Sons, New York, 2000.
- [209] P. Rai and H. Daume, “Beam search based map estimates for the Indian buffet process,” in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, June 2011, pp. 705–712.
- [210] J. C. McCall and M. M. Trivedi, “Driver behavior and situation aware brake assistance for intelligent vehicles,” *Proceedings of the IEEE*, vol. 95, no. 2, pp. 374–387, Feb 2007.
- [211] J. Wang, C. Yu, S. E. Li, and L. Wang, “A forward collision warning algorithm with adaptation to driver behaviors,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1157–1167, Apr. 2016.
- [212] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? The KITTI vision benchmark suite,” in *Proceedings of the 25th IEEE International Conference on Computer Vision and Pattern Recognition*, Tsukuba, Japan, Nov. 2012.
- [213] K. B. Petersen and M. S. Pedersen, “The matrix cookbook,” Nov. 2012, Version 20121115.

Curriculum Vitae

Name: Jürgen Hahn
Date of birth: 29. Dezember 1985
Place of birth: Berlin-Friedrichshain

Education

11/2009–08/2011 Technische Universität Darmstadt, Germany
Information Systems Technology, Master of Science
10/2006–10/2009 Technische Universität Darmstadt, Germany
Information Systems Technology, Bachelor of Science
07/2005 High school degree (Abitur)

Work experience

10/2011–09/2016 Research associate
Signal Processing Group
Institut für Nachrichtentechnik
Technische Universität Darmstadt, Germany
05/2015 Visiting researcher
Visual and Audio Signal Processing Lab
ICT Research Institute
University of Wollongong, Wollongong, NSW, Australia
03/2010–03/2011 Student research associate
Visual Computing for Medicine
Fraunhofer Institute for Computer Graphics Research
IGD, Darmstadt, Germany
07/2008–08/2008 Internship
Embedded Software Engineering
Fraunhofer Institute for Experimental Software Engineering IESE, Kaiserslautern, Germany

Erklärung laut §9 der Promotionsordnung

Ich versichere hiermit, dass ich die vorliegende Dissertation allein und nur unter Verwendung der angegebenen Literatur verfasst habe. Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

Darmstadt, 13. September 2016,

