TECHNISCHE
UNIVERSITÄT
DARMSTADT

HALF-QUADRATIC INFERENCE AND LEARNING FOR NATURAL IMAGES

Dissertation approved by
TECHNISCHE UNIVERSITÄT DARMSTADT
Fachbereich Informatik

for the degree of
Doktor-Ingenieur (Dr.-Ing.)

by

UWE SCHMIDT
M.Sc.

born in Hanau, Germany

Examiner:	Prof. Stefan Roth, Ph.D.
Co-examiner:	Prof. Paolo Favaro, Ph.D.
Date of Submission:	26th of October, 2016
Date of Defense:	16th of December, 2016

Darmstadt, 2017
D17

Uwe Schmidt: *Half-quadratic Inference and Learning for Natural Images*

ABSTRACT

MANY problems in computer vision are ill-posed in the sense that there is no unique solution without imposing additional regularization or prior knowledge about the desired result. In this dissertation, we are particularly interested in the *restoration* of *natural images*, which aims at recovering a clean image from a corrupted observation, such as an image afflicted by noise or blur.

In a *generative* approach, it is common to separate modeling of the image *prior* (regularization term) and the *likelihood* (data term), where the latter describes the mathematical relationship between the true image and its corrupted observation. By using Bayes' rule, prior and likelihood give rise to the *posterior* distribution of the restored image, which can then be used to infer the restored image. Alternatively, since prior and likelihood themselves are not actually needed to infer the restored image, the posterior can also be directly modeled in a *discriminative* approach.

The problem of *inference* is then to predict a restored image based on the posterior, where it is most common to seek the image with highest posterior probability. Inference typically involves solving an optimization problem of some kind, which can be difficult or slow, especially for non-convex optimization problems which often arise when trying to accurately model image restoration problems. To alleviate this issue, a particular optimization strategy known as *half-quadratic* (HQ) inference by Geman *et al.* [Geman and Reynolds, 1992; Geman and Yang, 1995] has proven to be very useful, where the model is first augmented with auxiliary variables. Inference then alternates between updating the restored image and the auxiliary variables, where both of these steps are relatively simple. Half-quadratic inference is a key component for all of the contributions put forward in this dissertation. Therefore, the first contribution is to provide a comprehensive review of HQ inference.

Our second contribution pertains to the issue that the likelihood often hinges on a few parameters (*e.g.*, the strength of assumed image noise), which are specific to the images at hand in a given application. Since these parameters are important but mostly unknown in practice, we address this (often ignored) issue by proposing a sampling-based inference method that allows to estimate such parameters besides the restored image. Half-quadratic inference plays an important role to make our approach practical.

Devising good image priors is often difficult, especially because natural images (and related scene types) have a complex structure. We address this throughout this thesis by using flexible images mod-

els based on *Markov random fields* (MRFs) and (parameter) *learning* based on example data. However, instead of hoping to learn a model that (approximately) adheres to some known regularities of the data, sometimes it is desirable to explicitly incorporate domain knowledge into the model. As our third contribution, we address this issue by enforcing invariance to linear transformations in a commonly-used class of models. With a focus on rotations, we propose transformation-aware feature learning and demonstrate our learned models in two applications. First, we learn an image prior that enables translation- and rotation-equivariant image denoising. Second, we devise rotation-equi-/invariant image descriptors based on learned rotation-aware features that perform well for rotation-invariant object recognition and detection.

In the following, we revisit and analyze HQ inference and propose an effective discriminative generalization based on a cascade of Gaussian *conditional random fields* (CRFs). By learning the model and its associated inference algorithm in a single unit, we show that using only few cascade stages yields excellent results in image denoising and deblurring. In particular, we propose the first discriminative non-blind deblurring approach that works for arbitrary images and blurs.

Finally, we address the issue that many low-level vision algorithms cannot be applied to megapixel-sized images. Based on our discriminative generalization of HQ inference, our final contribution is to learn a particularly efficient model and inference combination that can be applied to large images in a very reasonable amount of time, without compromising on the quality of the restored images.

ZUSAMMENFASSUNG

Viele Probleme in Computer Vision sind im mathematischen Sinne schlecht gestellt, d.h. es gibt keine eindeutige Lösung ohne das Problem zusätzlich zu regularisieren oder Vorwissen über die gewünschte Lösung einzubringen. Diese Dissertation beschäftigt sich hauptsächlich mit der *Restauration* von *natürlichen Bildern*, welche zum Ziel hat, ein fehlerloses Bild von einer fehlerhaften Beobachtung zu gewinnen, zum Beispiel von einem Bild das von Rauschen oder Unschärfe behaftet ist.

In einem *generativen* Ansatz ist es üblich, die Modellierung der *A-priori*-Wahrscheinlichkeit des Bildes (Regularisierungs-Term) und der *Likelihood* (Daten-Term) zu trennen, wobei die letztere den mathematischen Zusammenhang zwischen dem korrekten Bild und seiner fehlerhaften Beobachtung beschreibt. Aufgrund von A-priori-Wahrscheinlichkeit and Likelihood kann mit Hilfe des Satzes von Bayes die *A-posteriori*-Wahrscheinlichkeit gewonnen werden, aus welcher anschließend das restaurierte Bild geschätzt werden kann. Da A-priori-Wahrscheinlichkeit and Likelihood eigentlich nicht direkt zur Gewinnung des restaurierten Bildes benötigt werden, kann alternativ bei einem *diskriminativen* Ansatz die A-posteriori-Wahrscheinlichkeit auch direkt modelliert werden.

Das Problem der *Inferenz* ist nun ein restauriertes Bild mittels der A-posteriori-Wahrscheinlichkeit zu schätzen, wobei es meist üblich ist, das Bild mit der höchsten A-posteriori-Wahrscheinlichkeit zu ermitteln. Inferenz ist typischerweise mit dem Lösen eines Optimierungsproblems verbunden, was sich als schwierig oder langsam herausstellen kann, vor allem für nicht-konvexe Optimierungsprobleme, welche oft bei der sorgfältigen Modellierung von Bildrestaurierungsproblemen auftreten. Um dieses Problem zu mindern hat sich eine gewisse Optimierungsstrategie von Geman *et al.* [Geman and Reynolds, 1992; Geman and Yang, 1995], bekannt als *halb-quadratische* (HQ) Inferenz, als besonders nützlich herausgestellt, wobei das Modell zu Beginn mit zusätzlichen Hilfsvariablen ausgestattet wird. Inferenz wird nun durch das alternierende Anpassen des Bildes und der Hilfsvariablen durchgeführt, wobei jeder dieser beiden Schritte relativ einfach durchzuführen ist. Halb-quadratische Inferenz ist eine Kernkomponente für alle in dieser Dissertation vorgestellten wissenschaftlichen Beiträge. Daher ist der erste Beitrag eine umfassende Übersicht zur HQ Inferenz.

Unser zweiter Beitrag betrifft die Tatsache dass die Likelihood oft von einigen Parametern abhängt, welche jedoch spezifisch für die konkreten Bilder in einer gegebenen Anwendung sind. Da diese Pa-

rameter wichtig, aber praktisch meist unbekannt sind, adressieren wir dieses (oft ignorierte) Problem durch eine Stichproben-basierte Inferenz-Methode, die es erlaubt, solche Parameter neben dem restaurierten Bild zu schätzen. Halb-quadratische Inferenz spielt dabei eine wichtige Rolle, um unseren Ansatz zweckmäßig zu machen.

Gute A-priori-Wahrscheinlichkeiten für Bilder zu entwickeln ist oft nicht einfach, insbesondere da natürliche Bilder (und ähnliche Arten von Szenen) eine komplexe Struktur besitzen. Wir befassen uns in dieser Arbeit durchgehend mit dieser Problematik, indem wir flexible Bild-Modelle basierend auf *Markov random fields* (MRFs) und das *Lernen* von Parametern mittels Beispiel-Daten, verwenden. Anstatt jedoch zu hoffen, dass ein gelerntes Modell gewisse Regularitäten der Daten (approximativ) festhält, ist es manchmal wünschenswert, Domänenwissen explizit in das Modell einfließen zu lassen. Als unseren dritten Beitrag behandeln wir diese Thematik, indem wir Invarianz bezüglich linearen Transformationen in einer oft verwendeten Klasse von Modellen erzwingen. Mit einem Schwerpunkt auf Rotationen, schlagen wir transformations-bewusstes Lernen von Merkmalen vor und demonstrieren unsere gelernten Modelle in zwei Anwendungen. Zuerst lernen wir eine A-priori-Wahrscheinlichkeit von Bildern, welche translations- und rotations-equivariantes Bildentrauschen ermöglicht. Als zweites entwickeln wir rotations-equi-/invariante Bilddeskriptoren basierend auf rotations-bewusst gelernten Merkmalen, welche gute Ergebnisse für rotations-invariante Objekterkennung und -detektion liefern.

Anschließend greifen wir HQ Inferenz wieder auf, durch dessen Analyse wir zu einer effektiven diskriminativen Generalisierung gelangen, die durch eine Kaskade von Gaussian *conditional random fields* (CRFs) realisiert wird. Indem wir das Modell und den zugehörigen Inferenz-Algorithmus vereinen und gemeinsam lernen, zeigen wir dass nur wenige Stufen einer Kaskade ausreichen, um exzellente Ergebnisse im Entfernen von Bildrauschen und -unschärfe zu erzielen. Konkret entwerfen wir den ersten diskriminativen Ansatz für das nicht-blinde Entfernen von Bildunschärfe, welcher für beliebige Bilder und Unschärfen geeignet ist.

Letztlich widmen wir uns dem Thema, dass viele Algorithmen in "low-level" Computer Vision nicht auf Bilder in Megapixel-Größe anwendbar sind. Basierend auf unserer diskriminativen Generalisierung von HQ Inferenz, ist unserer letzter Beitrag das Lernen einer besonders effizienten Kombination aus Modell und Inferenz, welche auf große Bilder in sehr annehmbarer Zeit angewendet werden kann, ohne dabei die Qualität der restaurierten Bilder zu beeinträchtigen.

ACKNOWLEDGMENTS

I consider myself to have been a rather lucky Ph.D. student, which includes all the people I had the privilege to interact with. First and foremost, I want to thank my supervisor Stefan Roth, from whom I learned many things. Furthermore, I am grateful for having been surrounded by many nice colleagues over the years, especially my office mates and regular lunch buddies: Anton Milan, Jochen Gast, Kevin Schelten, Qi Gao, Stephan Richter, Thorsten Franzel, and Tobias Plötz. Moreover, I enjoyed interactions with the research groups of Bernt Schiele and Michael Goesele, in particular during retreats or at conferences. I experienced excellent working conditions, for which I partly have to thank the administrative and technical support at GRIS, especially Nils Balke for quickly solving technical issues.

I have been fortunate to visit or work with people outside Darmstadt during my Ph.D., such as the IUPR group at the University of Kaiserslautern, Smiths Heimann in Wiesbaden, the machine learning and perception group at Microsoft Research in Cambridge (UK), and the computer vision lab at TU Dresden. Especially the internship at Microsoft Research has been influential for my research. During my three months in Cambridge, I met a bunch of extraordinary people. I especially thank Carsten Rother, Sebastian Nowozin, Pushmeet Kohli, and Jeremy Jancsary for good collaboration during my time there and beyond. Moreover, I had fun and good discussions (typically while playing pool) with my fellow interns, especially Theofanis Karaletsos, Peter Kontschieder, and Andreas Müller. I enjoyed my short visit with Carsten Rother at TU Dresden, where I also had a good time with Shuai (Kyle) Zheng and Michael Hornáček, who were visiting at the time, too. I especially thank Michael for proofreading parts of this dissertation.

Finally, I deeply appreciate my friends and family that help me have a life outside the office. I am especially grateful to my girlfriend Daniela. Thank you for all the support and understanding over the years. Last but not least, I want to thank my parents who never pressured me to take a specific path in life. They supported my education and let me follow my interests, and for that I am truly thankful.

CONTENTS

1 INTRODUCTION 1
  1.1 Challenges 2
  1.2 Image Restoration 4
    1.2.1 Image denoising 5
    1.2.2 Image deblurring 6
  1.3 Bayesian Image Restoration 8
  1.4 Learning 9
  1.5 Half-quadratic Inference 10
  1.6 Thesis Overview 12
    1.6.1 Contributions 13
2 BACKGROUND AND RELATED WORK 15
  2.1 Probabilistic Graphical Models 15
    2.1.1 Bayesian networks 16
    2.1.2 Markov random fields 18
  2.2 MRF Image Models 20
    2.2.1 Pairwise MRFs and image statistics 20
    2.2.2 Fields of experts 23
    2.2.3 Posterior distribution 24
  2.3 Probabilistic Inference and Learning 25
    2.3.1 Bayesian point estimation 26
    2.3.2 Maximum likelihood learning 29
  2.4 Deterministic Inference and Learning 32
    2.4.1 Loss-based training 32
  2.5 Comparison of Modeling Approaches 34
    2.5.1 Generative 34
    2.5.2 Discriminative 35
    2.5.3 Discussion 37
  2.6 Image Restoration 37
    2.6.1 Denoising and deblurring 37
    2.6.2 Evaluation 39
    2.6.3 Other related work 42
3 HALF-QUADRATIC INFERENCE 45
  3.1 Gaussian MRFs 46
  3.2 Half-quadratic Augmentation 47
  3.3 Envelope and Integral Type 49
    3.3.1 Envelope type 49
    3.3.2 Integral type 51
    3.3.3 Summary 55
  3.4 Additive and Multiplicative Form 56
    3.4.1 Multiplicative form 58
    3.4.2 Additive form 67
    3.4.3 Summary 77

3.5 Solving Equation Systems 78
3.5.1 Matrix factorizations 80
3.5.2 Iterative solvers and preconditioners 83
3.5.3 Sampling 86
4 BAYESIAN IMAGE RESTORATION WITH UNOBSERVED VARIABLES 89
4.1 Estimating Unknown Parameters 91
4.2 Image Restoration 92
4.2.1 Half-quadratic MRF prior 93
4.3 Posterior Prediction 94
4.4 Bayesian Restoration using Sampling 96
4.5 Noise Estimation 98
4.6 Parametric Blur Estimation 99
4.7 Experiments 101
4.8 Summary 105
5 LEARNING ROTATION-AWARE FEATURES: FROM INVARIANT PRIORS TO EQUIVARIANT DESCRIPTORS 111
5.1 Product Models & Linear Transformations 113
5.1.1 Integrating transformation invariance 114
5.2 Learning Rotation-Invariant Image Priors 115
5.3 Learning Rotation-Aware Image Features 117
5.4 Rotation In-/Equivariant Image Descriptor 119
5.5 Experiments 123
5.6 Summary 126
6 CASCADES OF GAUSSIAN CONDITIONAL RANDOM FIELDS 129
6.1 Generalizing Half-quadratic Regularization 132
6.1.1 Discriminative alternative 133
6.1.2 Discriminative model cascade 135
6.2 Gaussian CRF for Restoration 136
6.2.1 Regression tree field 138
6.2.2 Training 139
6.2.3 RTF prediction cascade 140
6.3 Experiments 142
6.3.1 Image denoising 142
6.3.2 Image deblurring 146
6.4 Discussion 150
6.4.1 Training dataset 150
6.4.2 Model connectivity and comparison 151
6.5 Summary 153
7 DEEP SHRINKAGE FIELDS FOR EFFECTIVE IMAGE RESTORATION 155
7.1 Related work 156
7.2 Half-quadratic Baseline 157
7.2.1 Shrinkage function 160
7.3 Cascade of Shrinkage Fields 162
7.3.1 Learning 163

- 7.4 Experiments 165
- 7.5 Summary 170

8 SUMMARY AND FUTURE WORK 175
- 8.1 Contributions 175
  - 8.1.1 Generative models 175
  - 8.1.2 Discriminative models 177
- 8.2 Discussion and Outlook 178
  - 8.2.1 Generative and discriminative approaches 178
  - 8.2.2 Half-quadratic sampling 179
  - 8.2.3 Jointly learning model and inference 180
  - 8.2.4 Towards richer and more accurate models 182

A APPENDIX 185
- A.1 Supplemental Material for Chapter 5 185
  - A.1.1 Parameter learning in transformation-invariant product models 185
  - A.1.2 Details of parameter learning 186
  - A.1.3 Details of feature extraction 187
  - A.1.4 Additional descriptor details 188
  - A.1.5 Experimental details 189
- A.2 Supplemental Material for Chapter 7 190
  - A.2.1 Boundary handling 190
  - A.2.2 Learning 192

BIBLIOGRAPHY 197

LIST OF FIGURES

Figure 1.1 Comparison of real and synthetic image noise. 6
Figure 1.2 Common types of image blur. 7
Figure 1.3 Comparison of uniform and non-uniform blur. 8
Figure 2.1 Comparison of directed and undirected graphical models. 17
Figure 2.2 Two factor graphs for one graphical model. 19
Figure 2.3 Image statistics and potential functions. 22
Figure 2.4 Factor graph for image denoising with pairwise MRF. 24
Figure 3.1 Factor graph for augmented HQ image prior. 48
Figure 3.2 Multiplicative HQ form (example). 59
Figure 3.3 Update functions for MAP estimation with multiplicative HQ form. 62
Figure 3.4 Lower bound for both HQ types in multiplicative form (example). 64
Figure 3.5 Additive HQ form (example). 68
Figure 3.6 Updates functions for MAP estimation with additive HQ form. 70
Figure 3.7 Lower bound for both HQ types in additive form (example). 72
Figure 4.1 Deblurring results for various assumed Gaussian noise levels. 90
Figure 4.2 Visual comparison of noise estimation errors. 104
Figure 4.3 Deblurring example. 106
Figure 4.4 Deblurring example. 107
Figure 4.5 Blind deconvolution examples. 108
Figure 4.6 Blind deconvolution examples. 109
Figure 5.1 Rotation invariance and equivariance. 112
Figure 5.2 Learned R-FoE model (2 experts, 4 rotations). 116
Figure 5.3 Translation- and rotation-aware image features. 118
Figure 5.4 Simplified EHOF/IHOF descriptor example. 121
Figure 5.5 Denoising example (cropped). 123
Figure 5.6 Aerial car detection. 128
Figure 6.1 Standard HQ vs. discriminative cascade. 135
Figure 6.2 Examples of artificially generated blur kernels. 139
Figure 6.3 RTF prediction cascade (deblurring). 140
Figure 6.4 Image denoising example (cropped). 144
Figure 6.5 Color denoising example (cropped). 145
Figure 6.6 Deblurring example (cropped). 147
Figure 6.7 Realistic deblurring example. 148
Figure 6.8 Deblurring example at different model stages. 152

Figure 6.9 Factor types for 8-connected random fields. 153
Figure 7.1 Penalty functions, quadratic approximations, and shrinkage functions. 158
Figure 7.2 First two stages of learned $\text{CSF}_{3\times3}$ model. 165
Figure 7.3 Runtime comparison for image denoising. 169
Figure 7.4 Denoising example ($\sigma = 25$, cropped). 172
Figure 7.5 Deconvolution example (cropped). 173

LIST OF TABLES

Table 3.1 Overview of the four HQ variants. 56
Table 4.1 Average results for denoising and noise estimation. 102
Table 4.2 Comparison of average deblurring results. 103
Table 4.3 Noise estimation results. 104
Table 5.1 Test error on MNIST and MNIST-rot. 125
Table 6.1 Denoising results ($\sigma = 25$, no quantization). 142
Table 6.2 Denoising results ($\sigma = 25$). 142
Table 6.3 Color denoising results ($\sigma = 25$). 143
Table 6.4 Average deblurring results for 64 images from [Schmidt et al., 2011]. 146
Table 6.5 Average deblurring results for 32 images from [Levin et al., 2011]. 150
Table 6.6 Performance gain over the results of [Xu and Jia, 2010] in the benchmark of [Köhler et al., 2012]. 151
Table 7.1 Denoising results ($\sigma = 25$). 166
Table 7.2 Denoising results ($\sigma = 15, 25$, no quantization). 167
Table 7.3 Deconvolution results (using various kernel estimates). 170

ACRONYMS

BCCB	block circulant with circulant blocks
BM3D	block-matching and 3D filtering [Dabov et al., 2007b]
BSDS	Berkeley segmentation dataset [Arbelaez et al., 2011]
CG	conjugate gradient [Hestenes and Stiefel, 1952]
CPU	central processing unit
CRF	conditional random field [Lafferty et al., 2001]
DFT	discrete Fourier transform
DSP	digital signal processor
EM	expectation maximization [Dempster et al., 1977]
FoE	Field of Experts [Roth and Black, 2009]
GLM	Gaussian location mixture
GMM	Gaussian mixture model
GM	graphical model
GPU	graphics processing unit
GSM	Gaussian scale mixture
GT	ground truth
HQ	half-quadratic
i.i.d.	independent and identically distributed
LHS	left-hand side
MAP	maximum a-posteriori
MATLAB	`http://www.mathworks.com/products/matlab/`
MMSE	minimum mean squared error
MRF	Markov random field
PSNR	peak signal-to-noise ratio
RHS	right-hand side
SPD	symmetric positive-definite
SSIM	structural similarity [Wang et al., 2004]
w.r.t.	with respect to

1

INTRODUCTION

CONTENTS

1.1 Challenges 2
1.2 Image Restoration 4
  1.2.1 Image denoising 5
  1.2.2 Image deblurring 6
1.3 Bayesian Image Restoration 8
1.4 Learning 9
1.5 Half-quadratic Inference 10
1.6 Thesis Overview 12
  1.6.1 Contributions 13

ELECTRONIC circuits are pervasive in our daily lives, especially as they have become ever smaller and cheaper to manufacture. As a result, digital sensors of all kinds – including cameras – are now ubiquitous. For example, many people carry cell phones with them every day, which are equipped with digital cameras. Cameras are also indispensable for experimental and exploratory research, where visual data is often crucial to making sense of observed phenomena or experiments. Images and videos also play a major role in business, with applications ranging from surveillance to quality control.

However, the raw image data is just a means to an end, namely to find and explain patterns in the data, often with the ultimate goal of taking action based on understanding gained from it. Interpretation of visual data (in real time) is an essential skill for many animals as well as humans to act in the three-dimensional (3D) world around them, often ultimately necessary for survival. As a result, people are very good at explaining images and videos.

However, since the rate is accelerating at which data is acquired and recorded, automated or computer-assisted understanding of data – including images and videos – is more relevant than ever. As a result, *computer vision*, whose goal is to devise algorithms that enable computers to interpret and act upon visual information, is a growing field of research. Computer vision is concerned with a diverse range of applications. This includes tasks that often seem easy to people but are difficult for computers, such as *recognizing* [Lazebnik et al., 2006; Krizhevsky et al., 2012] and *localizing* [Viola and Jones, 2001; Felzenszwalb et al., 2010] many different kinds of objects, or *tracking* [Reid, 1979; Okuma et al., 2004] them over time. Applications of this

Note that we do not restrict labels *to be discrete.*

kind are typically called *high-level* vision. In contrast, the focus of this dissertation is *low-level* vision, which is concerned with assigning a respective output or *label* to every pixel (or voxel) of the image (or video), something that humans are (consciously) not familiar with.

LOW-LEVEL VISION Although low-level vision is an important area of research in itself, the outputs of such methods can additionally serve as inputs to algorithms that are concerned with high-level vision problems (*cf*. Chapter 5). For example, given two images of a scene taken from slightly different viewpoints, *stereo matching* [Lucas and Kanade, 1981; Sun et al., 2003] aims to estimate the depth from a reference viewpoint for every pixel in the scene, which can be an important feature in other applications (*e.g.*, for grouping of pixels that belong to the same object). A related problem is that of *optical flow* [Horn and Schunck, 1981; Sun et al., 2010], which tries to estimate the (apparent) 2D motion of every pixel between successive frames of a video. Another application is *image segmentation* [Rother et al., 2004; Felzenszwalb and Huttenlocher, 2004], which can be used for separating background from foreground (objects), which is often a prerequisite for other tasks, such as an *intrinsic image* decomposition [Land and McCann, 1971; Gehler et al., 2011] into reflectance (material-dependent, including color) and shading layers. In turn, the shading component of an image can be used to estimate surface normals and thus 3D shape, which is known as *shape from shading* [*cf*. Zhang et al., 1999]. Finally, although image sensors continue to improve in quality and resolution, they will never be perfect. Additionally, image quality is constrained by the fact that only a limited number of photons can be captured by the sensor. Hence, some image corruption is inevitable while an image or video is acquired. Additionally, unwanted image artifacts can be caused by a multitude of other (sometimes unavoidable) reasons, such as storage or transmission. In this thesis, we will focus on applications that aim to remove such corruption, which are grouped under the umbrella term of *image restoration* [*cf*. Katsaggelos, 2012]; we will discuss these in more detail in Section 1.2. However, our proposed methods are often not limited to a particular application, *i.e.* they can conceivably be generalized to other low-level vision tasks.

1.1 CHALLENGES

Unfortunately, many problems in low-level vision are severely *under-constrained*, *i.e.* there are more unknown labels to estimate than constraints provided by the given images. Additionally, the observed images are often contaminated by (random) noise. As a result, it is hopeless to choose among all possible solutions without imposing some form of *regularization* based on prior knowledge about the result we expect.

To that end, a common approach is to first devise a mathematical model of the observed images (called *data term* or *likelihood*), based on our assumptions of how they arose. For example, in *image deconvolution*, it is assumed that the observed image was produced by convolving the original image with a blurring filter. Note that the data term often hinges on a few parameters (*e. g.*, the strength of assumed image noise), which are specific to the images at hand in a given application. Since these parameters are important but mostly unknown in practice, we address this (often ignored) issue in Chapter 4. When the data model is not sufficient to guarantee a unique and sensible solution, a regularization or *prior* term is additionally used to help choose among possible solutions. For example, in case of image restoration, we need to (mathematically) encode our prior knowledge about "good" images.

LEARNING Devising good regularization terms is often difficult, especially because natural images (and related scene types) have a complex structure. We address this throughout this thesis by using flexible image models and (parameter) *learning* based on example data. However, instead of *hoping* to learn a model that (approximately) adheres to some known regularities of the data, sometimes we want to explicitly incorporate domain knowledge into the model. We address this issue in Chapter 5, where we describe how to *enforce* invariance to linear transformations in a commonly-used class of models.

INFERENCE After data and regularization terms are fully specified, we need to carry out *inference* to find a solution that reflects both terms. Similarly, inference is also necessary for model learning. Unfortunately, inference can be difficult since suitable regularization terms often lead to demanding optimization problems. We address this by recasting a complicated inference problem as a sequence of easier ones, specifically involving well-understood quadratic optimization. To that end, we adopt and extend the *half-quadratic* inference approach by Geman *et al.* [Geman and Reynolds, 1992; Geman and Yang, 1995] throughout this dissertation. We provide an extensive review of half-quadratic inference and related topics in Chapter 3 and propose effective generalizations in Chapters 6 and 7.

LARGE-SCALE PROBLEMS Many low-level vision algorithms cannot be applied to megapixel-sized images, which are common nowadays. This is because the involved optimization problems for inference do not scale to millions of variables (*i. e.*, pixels). To address this, we propose an efficient model and inference combination in Chapter 7 that we jointly learn from example data and that can be applied to such large-scale images in a reasonable amount of time.

Although we propose probabilistic and deterministic approaches in this dissertation, we typically discuss all models from a probabilistic point of view. However, an entirely non-probabilistic exposition would often be possible in case we were to seek only the single most probable solution for a given application.

1.2 IMAGE RESTORATION

Since we focus on applications of half-quadratic models in image restoration, we will now review the principal challenges and foundations of many restoration approaches. Image restoration is important, since removing corruption and artifacts from images reduces undesirable visual appearance variations and thus may help to improve the results of computer vision algorithms at higher levels. Likewise, it is also desirable to improve the quality of images for further human visual inspection, or simply to enhance consumer photographs.

For some of our proposed methods, it is straightforward to extend them to color images. Most of them can also be applied separately for each color channel with reasonable results.

Even though many of the techniques developed here could be generalized to videos or three-dimensional (voxel) images, this dissertation considers only two-dimensional still images. Most (consumer) cameras record color images with three different *channels* for the colors red, green, and blue (*RGB color model*). Here, we mostly work with *grayscale images*, which contain only one channel that represents the intensity (brightness) of the scene, since the key challenges for color and grayscale images are typically the same. Furthermore, we assume that we are working with – what are often called – *natural images*, *i.e.* images depicting scenes of nature or human-made environments; loosely speaking, these are images that are typically taken with consumer cameras.

Image restoration is an umbrella term for a wide variety of applications that aim to remove image corruption. Common forms of image corruption include:

NOISE Analogously to acoustic noise, image noise generally refers to random visually disturbing artifacts that were not present in the scene when the image was recorded.

BLUR Image blur denotes the condition that a single pixel of the image is a combination of several distinct points of a recorded scene.

OPTICAL ABERRATIONS Cameras typically include several optical elements (lenses) to record a sharp image of a scene. Optical aberrations are related to image blur and can be caused by a multitude of issues that arise in reproducing a sharp image of the scene due to (mis-)alignment or suboptimal quality of optical elements.

COMPRESSION (LOSSY) Images are often compressed, *e.g.* for storage or transmission purposes. This ranges from simply reducing the spatial resolution of the image to applying more sophisticated lossy compression schemes, such as the widely used *JPEG compression* [*cf.* Wallace, 1991].

In this thesis, we focus on removing noise and blur from natural images. However, many of the proposed methods can conceivably be adapted to other image domains or restoration applications.

1.2.1 *Image denoising*

Image noise denotes a random deviation of the image signal (*e.g.*, brightness, color) from its ideal value. Image denoising is thus concerned with recovering the true image signal from the observed noisy image. There are several sources of noise in modern digital cameras. Some noise arises due to the various electronic circuits that are used to record the image signal. Since the image sensor essentially counts photons, the random arrival of photons at the sensor causes what is called *shot noise*. Furthermore, consumer cameras mostly use sensors with *color filter arrays* (often a *Bayer filter* [Bayer, 1976]), which require a reconstruction step (called *demosaicing*) to obtain color values for every image pixel; demosaicing can cause image artifacts [*e.g.*, Chatterjee et al., 2011]. Additionally, there is *quantization noise* due to each pixel of the image being quantized to a discrete value for storage and transmission (commonly 256 values per pixel and channel in standard image formats).

While shot noise can be modeled with a Poisson distribution, quantization noise adheres to a uniform distribution. Noise due to other electronic circuits is often assumed to follow a Gaussian (normal) distribution. However, most common in the literature [*e.g.*, Portilla et al., 2003; Dabov et al., 2007b; Roth and Black, 2009; Jain and Seung, 2009] is to model image noise *overall* with a single Gaussian distribution, independently for each pixel

$$y_i = x_i + r, \qquad r \sim \mathcal{N}(0, \sigma^2) \tag{1.1}$$

We typically denote two-dimensional images as vectors to simplify notation. Throughout this dissertation, $\mathbf{y}$ *always denotes the observed corrupted image, and* $\mathbf{x}$ *the desired restored image.*

of an observed noisy image $\mathbf{y} \in \mathbb{R}^n$ where σ^2 denotes the noise variance (larger value denotes stronger noise). Hence, $\mathbf{y}$ is assumed to be a combination of the true unknown image $\mathbf{x} \in \mathbb{R}^n$ with additive white Gaussian noise. Given the true image, this gives rise to expressing the probability of the observed image via a multivariate Gaussian distribution

$$p(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} \mathcal{N}(y_i; x_i, \sigma^2) = \mathcal{N}(\mathbf{y}; \mathbf{x}, \sigma^2 \mathbf{I}), \tag{1.2}$$

Note that $p(x)$ *is shorthand notation for* $p(X = x)$, i.e. *the probability of random variable* X *having value* x.

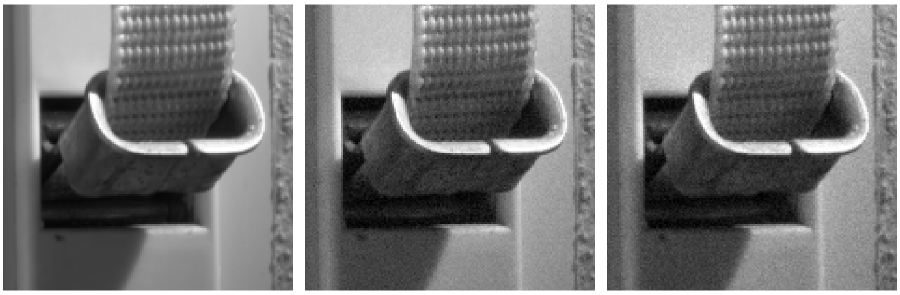

(a) Clean image (b) Real noisy image (c) Synthetic noisy image

Figure 1.1: **Noise comparison.** *(a)* Virtually noise-free image. *(b)* Image with real noise. *(c)* Synthetic noisy image based on *(a)* by adding white Gaussian noise of similar strength as in *(b)*. *Best viewed on screen.*

where **I** denotes the identity matrix. We choose to model pixels of an image with real numbers instead of enforcing discrete values (*e.g.*, $0, \ldots, 255$), since this generally makes modeling and inference easier.

The likelihood is a mathematical specification of the image corruption and thus application-dependent.

The distribution in Eq. (1.2) is called the *likelihood* function of the observed noisy image $\mathbf{y}$, given the noise-free image $\mathbf{x}$. We also make a Gaussian noise assumption for most of the methods in this dissertation.

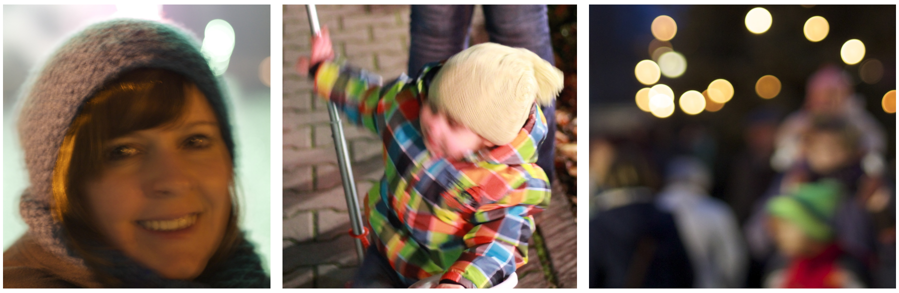
A Gaussian noise assumption is also important for the half-quadratic models that we consider in this thesis (Chapter 3).

There is some justification from the (Liapunov) *central limit theorem* for modeling the observed image with a Gaussian distribution, since the observed noise in an image is assumed to be accumulated from various independent sources (*e.g.*, shot noise and quantization noise). However, making a Gaussian noise assumption is also mathematically very convenient, which can be argued to at least partly account for its widespread use. Furthermore, the literature on image denoising rarely addresses the removal of noise from real photographs [a recent exception: Anaya and Barbu, 2014]. Instead, it is common to denoise artificially created images to facilitate quantitative comparisons to other denoising methods. Since those artificial test images are typically randomly simulated according to Eq. (1.2), the Gaussian noise assumption obviously holds. Figure 1.1 shows a comparison between real and synthetic noise for an example image.

It can be argued that image denoising (under the assumption of Gaussian noise) has become a benchmark to compare various generic image models, which are also applicable to other tasks. This might be the case due to image denoising being one of the easiest image restoration tasks.

1.2.2 *Image deblurring*

Although image blur can be used for artistic effect (*e.g.*, *bokeh*), it is often unpleasant and severely reduces the sharpness of an image.



(a) Camera shake (b) Object motion blur (c) Defocus blur

Figure 1.2: **Examples of common types of image blur.**

There are various common causes of image blur with different manifestations, especially:

CAMERA MOTION The whole image is blurred if the camera is moved during exposure. This very common cause of blur is also called camera shake (Fig. 1.2(a)), since it is often due to shaking the camera while holding it, in particular with slower shutter speeds.

OBJECT MOTION Only parts of the image become blurred if objects move while the shutter is open during exposure (Fig. 1.2(b)).

DEFOCUS Defocus blur refers to the recorded image being out-of-focus (Fig. 1.2(c)), which can be caused by an unsuitable arrangement of the camera (lenses) or if the objects of interest changed their distance to the camera.

In image deblurring, it is common to model a pixel

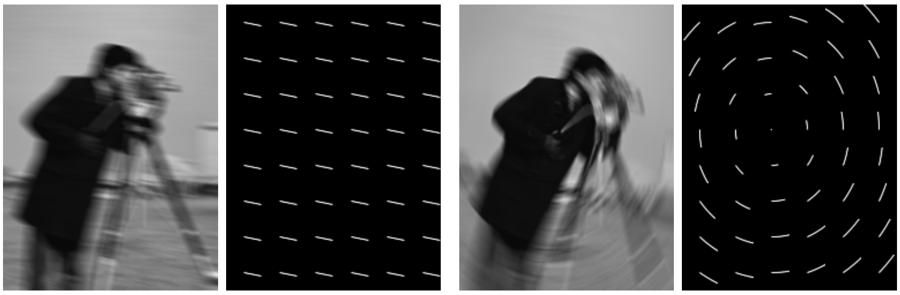
$$y_i = \left(\sum_{j=1}^{m} K_{ij} x_j\right) + r \tag{1.3}$$

of the observed blurred image $\mathbf{y} \in \mathbb{R}^n$ by means of a *blur matrix* $\mathbf{K} \in \mathbb{R}^{n \times m}$ as a linear combination of the pixels of the underlying true image $\mathbf{x} \in \mathbb{R}^m$ plus some pixel-independent noise r. Again, we assume here $r \sim \mathcal{N}(0, \sigma^2)$ to follow a Gaussian distribution, which, given the true image, also allows us to express the probability of the whole observed image as a multivariate Gaussian:

Note that $\mathbf{x}$ and $\mathbf{y}$ can have different sizes.

$$p(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} \mathcal{N}\left(y_i; \sum_{j=1}^{m} K_{ij} x_j, \sigma^2\right) = \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2 \mathbf{I}). \tag{1.4}$$

From the above definition of the deblurring likelihood in Eq. (1.4), it also becomes apparent that the denoising likelihood (Eq. 1.2) is retained as a special case when $\mathbf{K}$ is the identity matrix. However, in contrast to denoising we typically assume less noise (*i. e.*, smaller variance σ^2). Furthermore, image deblurring methods in the literature are frequently evaluated on real images to demonstrate their merits [*e. g.*, Fergus et al., 2006; Cho and Lee, 2009; Joshi et al., 2010].



(a) Uniform blur (b) Non-uniform blur

Figure 1.3: **Comparison of uniform and non-uniform blur.** The images on the right in both cases depict the blur kernel at selected locations.

The blur kernel is also called point spread function.

In addition, uniform blur can only be fully accurate when all pixels have the same depth.

The blur matrix $\mathbf{K}$ is typically very sparse since we assume that only relatively few pixels of the true unobserved image contribute to a single pixel of the observed blurred image. Furthermore, the blur is called *uniform* if it is the same at all locations of the image. In this case, the blur matrix $\mathbf{K}$ has special repetitive structure that allows to express multiplication as *convolution* with a *blur kernel* $\mathbf{k}$ that is much smaller than the image, *i. e.* $\mathbf{Kx} \equiv \mathbf{k} \otimes \mathbf{x}$ with $\mathbf{k} \in \mathbb{R}^p, p \ll m$. Hence, deblurring is also called *deconvolution* when assuming uniform blur.

Note that uniform blur is a somewhat strong assumption to make, since translation along the image plane is the only camera motion that can result in uniform blur. However, uniform blur is still the dominant assumption in the literature to simplify the deblurring problem [*e. g.*, Levin et al., 2009; Schuler et al., 2013]. Only recent work has started to make different assumptions (*e. g.*, 3D camera rotation [Whyte et al., 2010]), which result in *non-uniform* blur. Figure 1.3 shows a comparison between uniform and non-uniform blur for an example image. Note that results obtained with a uniform blur assumption can be quite reasonable in practice, and have been found to outperform some non-uniform blur methods even if the true blur is not (quite) uniform [Köhler et al., 2012]. While some of the proposed methods in this dissertation are compatible with both uniform and non-uniform blur assumptions, all our experiments are carried out in the context of uniform blur.

Similarly, blind *denoising refers to the case where the noise strength is unknown.*

The deblurring problem is typically called *non-blind* under the assumption that the blur matrix (or kernel) is known. In contrast, *blind* deblurring refers to the problem of estimating the blur *and* deblurring the image, although these two parts are often carried out separately [*cf.* Levin et al., 2009].

1.3 BAYESIAN IMAGE RESTORATION

Our discussion has so far been confined to why and how images get corrupted. Concretely, for image noise and blur, we used a likelihood function to model how we assume the corrupted image to be related to the original clean image. This is also called the *forward model*, since it encodes how we get from the original image to the corrupted one.

Image restoration is consequently called an *inverse problem*, since it essentially amounts to "inverting" the forward model. Unfortunately, this is mathematically *ill-posed* [Hadamard, 1923] since not sufficiently constrained or because the forward model is non-deterministic (due to the presence of noise). To remedy this, we can impose *regularization* [Tikhonov, 1963] to express our preference for certain manifestations of $\mathbf{x}$. To that end, we specify $p(\mathbf{x})$, called the *prior* distribution, since it encodes our prior knowledge of good images, independent of the particular application and before observing any corrupted image.

Separately modeling prior and likelihood, which also defines the joint distribution $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y}|\mathbf{x}) \cdot p(\mathbf{x})$, is called a *generative* approach. By using *Bayes' rule*, we obtain the *posterior* distribution

$$p(\mathbf{x}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}) \cdot p(\mathbf{x}) \tag{1.5}$$

of the restored image $\mathbf{x}$, given the observed corrupted image $\mathbf{y}$. Assuming a sensible prior distribution, the posterior is now well-posed and can be used to predict the restored image. To that end, the most common choice is $\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$, *i. e.* the image with the highest posterior probability (density), which is called the *maximum a-posteriori* (MAP) estimate.

In a generative approach, we model the joint distribution $p(\mathbf{x}, \mathbf{y})$ but only require the posterior $p(\mathbf{x}|\mathbf{y})$ to predict the restored image. The idea in a *discriminative approach* is to directly model the posterior, without separately specifying prior and likelihood. Generative and discriminative approaches both have their advantages and disadvantages, some of which will be discussed in the subsequent Chapter 2. We propose approaches of both kinds in this dissertation, which are discussed in detail in their respective chapters.

ENERGY MINIMIZATION The posterior is often alternatively defined via an *energy* (*i. e.*, cost) function $E(\mathbf{x}|\mathbf{y})$ as $p(\mathbf{x}|\mathbf{y}) \propto \exp(-E(\mathbf{x}|\mathbf{y}))$. Hence, MAP estimation corresponds to *energy minimization*:

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y}) = \arg\min_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}). \tag{1.6}$$

Note that no probabilistic interpretation is necessary for energy minimization, where the energy is composed of a *data term* (corresponding to the likelihood) and a *regularization term* (akin to the prior).

1.4 LEARNING

We briefly discussed the necessity of a prior distribution $p(\mathbf{x})$ in a generative image restoration approach. Devising priors that favor "good" images over "poor" ones is a challenging task. Instead of directly operating on the raw pixel values, image priors typically use derived image representations, called *features*, that are more relevant for the task and lead to better models. To devise good features, it has proven useful to study the (statistical) properties of the images of interest, here natural images. For example, one of the most salient features of natural images is *smoothness*, *i. e.* neighboring pixels mostly have similar brightness (or color) values [*e. g.*, Ruderman, 1994].

Exploiting smoothness (and other features) has been very successful, but also has its limitations. It has been shown [*e. g.*, Roth and Black, 2009] that image priors can be substantially improved by using features that, although inspired by image statistics, have many additional degrees of freedom (*i. e.*, parameters). Hence, we can define a prior $p(\mathbf{x}; \mathbf{\Theta})$ based on parameters $\mathbf{\Theta}$ that also determine the feature representation of the model. Since manually choosing such parameters is difficult and cumbersome, *learning* has become very attractive, *i. e.* using techniques from *machine learning* to automatically choose good parameters based on exemplary data. Furthermore, learned features (*e. g.*, patterns) that are generally suitable to model images can also be useful for other tasks, such as object *classification* or *detection* (Chapter 5).

Although we motivated parameter learning for image priors in a generative context, the general idea is also applicable to learning parameters of a posterior distribution $p(\mathbf{x}|\mathbf{y}; \mathbf{\Theta})$ in a discriminative approach.

RANDOM FIELDS In particular, all our models are posed within the framework of *Markov random fields* (MRFs) [*cf.* Li, 2001], which are used here to specify a (probabilistic) model of whole images by modeling only local image neighborhoods. Note that these local neighborhoods overlap and are modeled based on the statistics of natural images (*e. g.*, smoothness). MRFs assume that a single pixel, given its local neighborhood of pixels, is independent of all other pixels in the image (*Markov property*). When such models are adapted to directly model the posterior in a discriminative context, they are typically called *conditional random fields* (CRFs) [Lafferty et al., 2001]. Chapter 2 will introduce MRFs and CRFs in more detail.

1.5 HALF-QUADRATIC INFERENCE

Once prior and likelihood in a generative setting have been specified, we need to carry out posterior inference (*e. g.*, MAP estimation) to predict the restored image, which can be difficult depending on the particular properties of the posterior distribution.

Recall that we assume a Gaussian distribution for the likelihood (for image denoising and deblurring). If we are able to also model the prior with a Gaussian, then the posterior could also be derived as a Gaussian distribution of the restored image (due to self-*conjugacy* [Raiffa and Schlaifer, 1961]). A good reason to use Gaussian distributions is that they are well-understood, and inference is relatively easy (*cf.* Section 3.5). This is partly due to the availability of efficient methods for inference with *quadratic* functions, since $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \propto \exp(-E_{\mathcal{N}}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}))$ is defined via a quadratic (energy) function

$$E_{\mathcal{N}}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}). \tag{1.7}$$

However, the statistics of commonly-used natural image features preclude the use of Gaussian image priors [*cf.* Ruderman, 1994]. For example, although images are mostly smooth, this does not hold at object boundaries (*e. g.*, at the outline of a person in front of a background). Modeling such smoothness "outliers" requires more complex, non-Gaussian, distributions. Unfortunately, posterior inference (especially probabilistic inference) can be very difficult and inefficient for such distributions. Although general-purpose gradient-based techniques can be used for MAP estimation, they can be slow to converge. Furthermore, they can get stuck in local optima in case of non-convex energy functions.

Inference is also important for learning.

An alternative inference approach, which we adopt throughout this dissertation, is to define an *augmented* prior $p(\mathbf{x}, \mathbf{z})$ with auxiliary variables $\mathbf{z}$, such that inference with the augmented prior is easier, but in principle yields the same result as if we were using the original prior. In particular, inference with the augmented prior makes use of the conditional distributions $p(\mathbf{x}|\mathbf{z})$ and $p(\mathbf{z}|\mathbf{x})$, which are easier to work with by construction. Furthermore, such an augmentation is called *half-quadratic* (HQ) [Geman and Reynolds, 1992; Geman and Yang, 1995] if $p(\mathbf{x}|\mathbf{z})$ is a Gaussian distribution (or equivalently $E(\mathbf{x}|\mathbf{z})$ quadratic in $\mathbf{x}$). This will be discussed in detail in Chapter 3.

Based on the augmented prior, an augmented (also half-quadratic) posterior $p(\mathbf{x}, \mathbf{z}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}) \cdot p(\mathbf{x}, \mathbf{z})$ is used for inference, albeit (theoretically) guaranteeing identical results to using the original posterior. Inference alternates between updating $\mathbf{x}$ and $\mathbf{z}$ according to the conditional distributions

Note that HQ inference hinges on a Gaussian likelihood $p(\mathbf{y}|\mathbf{x})$, which we assume here (Eqs. 1.2 and 1.4).

$$p(\mathbf{x}|\mathbf{z}, \mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}) \cdot p(\mathbf{x}|\mathbf{z}) \propto \mathcal{N}(\mathbf{x}; \ldots) \tag{1.8}$$

$$p(\mathbf{z}|\mathbf{x}, \mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}) \cdot p(\mathbf{z}|\mathbf{x}) \propto \textstyle\prod_k p(z_k|\mathbf{x}), \tag{1.9}$$

which are both relatively easy to work with, since $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$ is a Gaussian distribution and $p(\mathbf{z}|\mathbf{x}, \mathbf{y})$ is a product of univariate distributions if auxiliary variables $\mathbf{z}$ are chosen to be independent (as is common).

MAP estimation with such an augmented posterior can be shown to correspond to a second-order optimization method applied to the original posterior (Chapter 3). However, in contrast to standard second-order methods, HQ augmentation can also be used for probabilistic inference beyond MAP estimation since it also allows to draw samples from the augmented posterior (Chapter 4).

HQ inference can also be used for drawing samples from an image prior.

1.6 THESIS OVERVIEW

Chapter 2 introduces probabilistic *graphical models*, and Markov random fields (MRFs) in particular, since all our proposed models are based on this framework. We continue with a discussion of inference and parameter learning in probabilistic and non-probabilistic contexts. We conclude the first part of the chapter by comparing generative and discriminative random field models, since approaches of both kinds are put forward in this thesis. We then turn to image restoration and discuss common evaluation criteria for the quality of restored images, before we briefly survey related work with a focus on methods that are based on other paradigms.

Chapter 3 provides a detailed overview of half-quadratic (HQ) inference for MRF models, which is used throughout this dissertation. To better understand our proposed models in Chapters 4–7, we study the advantages and disadvantages of different variants of HQ inference and their connections to other optimization methods. Our discussion ends with strategies for solving systems of linear equations, which forms the backbone of HQ inference (for both MAP estimation and sampling).

Our main contributions, which are outlined in more detail in the following section, pertain to both to generative (Chapters 4 and 5) and discriminative (Chapters 6 and 7) approaches: Chapter 4 is partly based on [Schmidt et al., 2011][1] and presents a probabilistic approach to image restoration that is especially suited to the case when parameters of the corruption model are unknown. Chapter 5 has been published as [Schmidt and Roth, 2012] and proposes a generic modeling framework that allows feature learning with invariances to given linear image transformations, such as translations and rotations. Based on [Schmidt et al., 2013, 2016], Chapter 6 presents our discriminative generalization of HQ inference, which performs image restoration with a cascade of Gaussian conditional random fields. Chapter 7 has been published as [Schmidt and Roth, 2014] and generalizes a specific very efficient variant of HQ inference and thus enables large-scale image restoration.

In Chapter 8, we conclude the dissertation and discuss promising avenues for future work. Finally, Appendix A provides further details regarding Chapters 5 and 7.

---

[1] NOTE ON CONTRIBUTION: Kevin Schelten and myself contributed equally to the publication [Schmidt, Schelten, and Roth, 2011]. My contribution to the paper was primarily to integrate noise estimation for image denoising and deblurring, as well as to conduct the experiments. Chapter 4 is partly based on this contribution, which is further extended to parametric blur estimation.

1.6.1 *Contributions*

REVIEW OF HALF-QUADRATIC INFERENCE Half-quadratic inference is an important approach to convert challenging optimization problems into a sequence of quadratic problems, which are then easier to solve. Due to its widespread use, there is a substantial body of literature spanning multiple research communities. Based on a unified notation, we provide a comprehensive review of half-quadratic (HQ) inference for MRF models in Chapter 3. Furthermore, we discuss connections to related optimization approaches to better understand HQ inference.

IMAGE RESTORATION WITH UNKNOWN PARAMETERS Even if the image corruption process is assumed to be known, the resulting likelihood model frequently depends on parameters, such as the variance of the assumed Gaussian noise (*cf.* Eqs. 1.2 and 1.4). Although the quality of the restored image often crucially hinges on an appropriate choice of such likelihood parameters, they are typically assumed to be known and it is often not addressed how they can be estimated from the corrupted image. In Chapter 4, we propose a Bayesian image restoration approach with integrated estimation of unknown parameters, which are treated as unobserved random variables. Based on good generative image priors, we use probabilistic inference via sampling with HQ representations, which allows for joint inference of the restored image and likelihood parameters. With a focus on noise estimation, we demonstrate the efficacy of our approach in the context of image deblurring and denoising with integrated parameter estimation of the noise and blur models.

ROTATION-AWARE FEATURE LEARNING Identifying suitable image features is a central challenge for many applications in computer vision. Due to the difficulty of this task, techniques for learning features directly from example data have recently received attention. Despite significant benefits, these learned features often have many fewer of the desired invariances or equivariances than their handcrafted counterparts. While translation in-/equivariance has been addressed, the issue of learning rotation-invariant or equivariant representations has hardly been explored. In Chapter 5, we describe a general framework for incorporating invariance to linear image transformations in product models for feature learning. A particular benefit is that our approach induces transformation-aware feature learning, *i. e.* it yields features that have a notion with which specific image transformation they are used. We focus our study on rotation in-/equivariance and show the advantages of our approach in learning rotation-invariant image priors and in building rotation-equivariant and invariant descriptors of learned features, which result in excellent performance for rotation-invariant object detection.

CASCADES OF GAUSSIAN CRFS Conditional random fields (CRFs) are popular discriminative models for computer vision and have been successfully applied also in the domain of image restoration, especially to image denoising. For image deblurring, however, discriminative approaches have been mostly lacking. We posit two reasons for this: First, the blur kernel is often known only at test time, requiring any discriminative approach to cope with considerable variability. Second, given this variability it is quite difficult to construct suitable features for discriminative prediction. We address these challenges in Chapter 6 by first showing a connection between half-quadratic inference for generative image priors and Gaussian CRFs. Based on this analysis, we then propose a generalization in form of a cascade model for image restoration that consists of a Gaussian CRF at each stage. Each stage of our cascade is semi-parametric, *i. e.* it depends on the instance-specific parameters of the restoration problem, such as the blur kernel. We train our model discriminatively with synthetically generated training data. Our experiments show that when applied to image deblurring, the proposed approach is efficient and yields state-of-the-art restoration quality on images corrupted with synthetic and real blur. Moreover, we demonstrate its suitability for image denoising, where we achieve competitive results for grayscale and color images.

DEEP SHRINKAGE FIELDS Many state-of-the-art image restoration approaches do not scale well to larger images, such as megapixel images common in the consumer segment. Computationally expensive optimization is often the culprit. While efficient alternatives exist, they have not reached the same level of image quality. Based on insights from Chapter 6, in Chapter 7 we develop an effective approach to image restoration that offers both computational efficiency and high restoration quality. To that end we propose *shrinkage fields*, a discriminative generalization of an efficient variant of HQ inference, which can be thought of as a random field-based architecture that combines the image model and the optimization algorithm in a single unit. The underlying shrinkage operation bears connections to wavelet approaches, but is used here in a random field context. Computational efficiency is achieved by construction through the use of convolutions and discrete Fourier transforms as the core components; high restoration quality is attained through discriminative training of all model parameters and the use of a deep model (cascade architecture). Unlike heavily engineered solutions, our learning approach can be adapted easily to different trade-offs between efficiency and image quality. We demonstrate state-of-the-art restoration results with high levels of computational efficiency, and significant speedup potential through inherent parallelism.

2

BACKGROUND AND RELATED WORK

CONTENTS

2.1 Probabilistic Graphical Models **15**
  2.1.1 Bayesian networks 16
  2.1.2 Markov random fields 18
2.2 MRF Image Models **20**
  2.2.1 Pairwise MRFs and image statistics 20
  2.2.2 Fields of experts 23
  2.2.3 Posterior distribution 24
2.3 Probabilistic Inference and Learning **25**
  2.3.1 Bayesian point estimation 26
  2.3.2 Maximum likelihood learning 29
2.4 Deterministic Inference and Learning **32**
  2.4.1 Loss-based training 32
2.5 Comparison of Modeling Approaches **34**
  2.5.1 Generative 34
  2.5.2 Discriminative 35
  2.5.3 Discussion 37
2.6 Image Restoration **37**
  2.6.1 Denoising and deblurring 37
  2.6.2 Evaluation 39
  2.6.3 Other related work 42

THIS chapter first reviews basic mathematical foundations of modeling probability distributions over many variables, which motivate all models in this dissertation. After introducing MRF image models, we present probabilistic and deterministic variants of inference and parameter learning. We conclude this first part with a discussion of the advantages and disadvantages of different modeling approaches. The second part of this chapter is devoted to image restoration. Concretely, we discuss common evaluation methodologies and criteria, before we briefly survey related work with a focus on other modeling paradigms that differ from ours.

2.1 PROBABILISTIC GRAPHICAL MODELS

Most image models proposed in this dissertation are essentially probability distributions over the domain of all images (of a given size).

Since this domain and thus the associated space of probability distributions is vast, we need to make some simplifying assumptions in the form of (conditional) *independences* between random variables (*i. e.*, pixels). Two (sets of) random variables x_1, x_2 are independent if $p(x_1, x_2) = p(x_1)p(x_2)$, which implies $p(x_1|x_2) = p(x_1)$ and $p(x_2|x_1) = p(x_2)$. Likewise, two (sets of) variables x_1, x_2 are conditionally independent if $p(x_1, x_2|x_3) = p(x_1|x_3)p(x_2|x_3)$ given variable (set) x_3.

Probabilistic *graphical models* (GMs) [*cf.* Koller and Friedman, 2009] are very useful for defining probability distributions over many variables because they allow for encoding (conditional) dependencies and independences between variables in a principled way by means of a graph structure. In a GM, each random variable is represented by a *node* (vertex) in the graph. By convention, an observed random variable with known value is depicted with a shaded (*i. e.*, gray) node. Non-shaded (*i. e.*, white) nodes correspond to unobserved (called latent) random variables. The *edges* in the graph indicate the dependencies between the variables that are represented by the nodes. Since the semantics of the graph are agreed upon in the community, algorithms for inference and learning in GMs can directly make use of them.

Note that we will interchangeably refer to a random variable or its associated node in the graph.

Regarding the kinds of graphs and edges, there are two main families of GMs: *directed* and *undirected* ones. We will introduce both in the following; let $\mathbf{x} = [x_1, \ldots, x_D]^\mathsf{T}$ denote a random vector of D variables with $\{\mathbf{x}\} = \{x_1, \ldots, x_D\}$ representing the set of all variables.

2.1.1 *Bayesian networks*

Every probability distribution of a random vector $\mathbf{x} = [x_1, \ldots, x_D]^\mathsf{T}$ can be decomposed as a product of D conditional distributions as

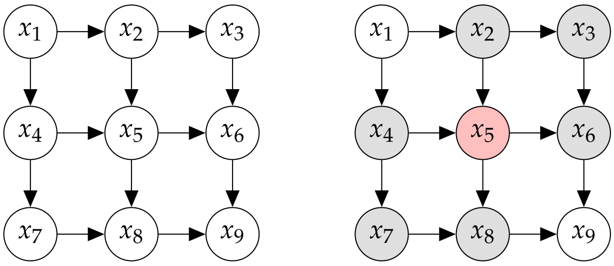
$$p(\mathbf{x}) = p(x_1)p(x_2|x_1)\cdots p(x_D|x_1, x_2, \ldots, x_{D-1}). \tag{2.1}$$

Since these conditionals depend on up to $D-1$ other variables, one approach to simplify this is to drop many of these dependencies. This idea is at the heart of directed GMs, which are also called *Bayesian networks*. Bayesian networks are represented by directed acyclic graphs (DAGs), *i. e.* all edges have a direction and there are no (directed) cycles in the graph. Given a directed edge from node u to v, u is called a *parent* of v, and v is consequently a *child* of u. The probability (density) of a random vector $\mathbf{x}$ is defined as the product of the conditional distributions of all variables, given their parents:

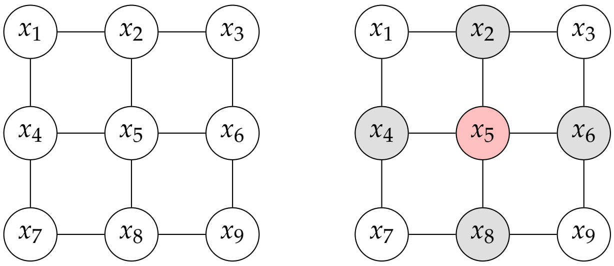
Since our discussion applies to both, note that we do not formally distinguish between probability mass *and* density *functions.*

$$p(\mathbf{x}) = \prod_{i=1}^{D} p(x_i|\mathrm{parents}(x_i)). \tag{2.2}$$

Note that $p(\mathbf{x})$ is a properly normalized distribution, since we assume that all conditionals are normalized.



(a) Directed GM (Bayesian network)



(b) Undirected GM (Markov random field)

Figure 2.1: **Examples of graphical models.** The graph structure is shown on the left, whereas the right part in both cases depicts the Markov blanket (shaded gray) for node x_5 (shaded red).

An important concept is the *Markov blanket* $\mathcal{M}(x_i)$ of a node x_i, since it denotes the minimal set of nodes that makes x_i conditionally independent of all other nodes in the GM, *i.e.* $p(x_i|\{\mathbf{x}\}\backslash\{x_i\}) = p(x_i|\mathcal{M}(x_i))$. Markov blankets also generalize to subsets of nodes and can always be inferred from the graph structure. In a directed GM, it can be shown [*cf.* Koller and Friedman, 2009, § 4.5] that the Markov blanket

$$\mathcal{M}(x_i) = \text{parents}(x_i) \cup \text{children}(x_i) \cup \text{parents}(\text{children}(x_i)) \tag{2.3}$$

of node x_i consists of its parents, children, and the parents of its children. Markov blankets are important to interpret the (simplifying) independence assumptions of the model and are also often exploited for inference in GMs.

EXAMPLE Assume a random vector $\mathbf{x} = [x_1, \ldots, x_9]^\mathsf{T}$ with $D = 9$ variables, which may be thought of as representing the pixels of an image of height and width 3.

Making the simplifying assumption that a pixel is conditionally independent of all others given a small neighborhood, the left part of Fig. 2.1(a) depicts a sensible directed GM [*cf.* Domke et al., 2008], which corresponds to the distribution

$$\begin{aligned} p(\mathbf{x}) = p(x_1)p(x_2|x_1)p(x_3|x_2)p(x_4|x_1)p(x_5|x_2,x_4)\cdot \\ p(x_6|x_3,x_5)p(x_7|x_4)p(x_8|x_5,x_7)p(x_9|x_6,x_8). \end{aligned} \tag{2.4}$$

In particular, given the concept of the Markov blanket, it can easily be verified that the "central" node x_5 is conditionally independent of all others, given 6 observed neighboring nodes (shaded gray in the right part of Fig. 2.1(a)):

$$p(x_5|\{\mathbf{x}\}\backslash\{x_5\}) = p(x_5|x_2, x_3, x_4, x_5, x_6, x_7, x_8). \tag{2.5}$$

Note that this holds true even when modeling images of larger sizes.

2.1.2 *Markov random fields*

Undirected GMs are also called *Markov random fields* (MRFs) [Besag, 1974; Geman and Geman, 1984] or *Markov networks*; they are represented with graphs that contain only undirected edges. Nodes in the graph are called *neighbors* if they are directly connected via an (undirected) edge. We denote by $\mathcal{C}$ the set of all *cliques* of the graph, which are subsets of nodes such that there is an edge between all pairs of distinct nodes in a clique; $\mathbf{x}_{(c)}$ indicates the subset of nodes that belong to a given clique $c \in \mathcal{C}$. Furthermore, we define a *factor* or (clique) *potential* φ_c as a non-negative function that assigns a real number to a particular configuration of nodes $\mathbf{x}_{(c)}$, *i. e.* $\varphi_c(\mathbf{x}_{(c)}) \geq 0$; this value can be thought of as a compatibility score or unnormalized probability (density).

A clique can consist of a single node.

Note that in the literature, $\log \varphi_c$ *is often referred to as "potential" instead.*

The Markov blanket of a node x_i in an undirected GM is simply the set of its neighbors, *i. e.* $p(x_i|\{\mathbf{x}\}\backslash\{x_i\}) = p(x_i|\mathcal{M}(x_i))$ with

$$\mathcal{M}(x_i) = \text{neighbors}(x_i). \tag{2.6}$$

The *Hammersley-Clifford theorem* [Hammersley and Clifford, 1971] now states that all distributions that satisfy these conditional independences (as implied by the graph structure) can be represented only as products of *positive* clique potentials, *i. e.*

$$p(\mathbf{x}) = \frac{1}{Z}\prod_{c\in\mathcal{C}} \varphi_c(\mathbf{x}_{(c)}). \tag{2.7}$$

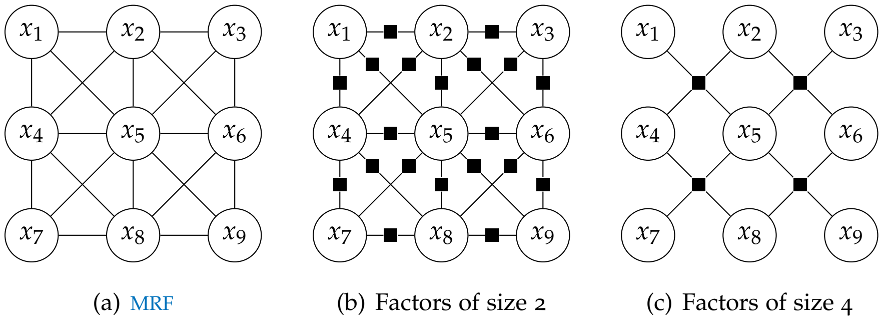
A normalization constant $Z = \int \prod_{c\in\mathcal{C}} \varphi_c(\mathbf{x}_{(c)})\, d\mathbf{x}$ is necessary to ensure that Eq. (2.7) is a valid probability distribution. Such MRFs with positive clique potentials ($\varphi_c(\mathbf{x}_{(c)}) > 0$) are also called *Gibbs distributions*. All models put forward in this thesis are Gibbs distributions, which are often equivalently defined as $p(\mathbf{x}) = \frac{1}{Z}\exp(-E(\mathbf{x}))$ via an associated *energy* function

Integration is replaced by summation for discrete $\mathbf{x}$.

Note that we always denote by log *the* natural logarithm.

$$E(\mathbf{x}) = -\sum_{c\in\mathcal{C}} \log \varphi_c(\mathbf{x}_{(c)}). \tag{2.8}$$

An MRF is called *pairwise* if all cliques $c \in \mathcal{C}$ contain at most two nodes, whereas cliques of three and more nodes give rise to *high-order* MRFs.



(a) MRF (b) Factors of size 2 (c) Factors of size 4

Figure 2.2: Factor graphs with factors of different sizes *(b,c)* that both represent the same undirected graphical model *(a)*.

EXAMPLE Let $\mathbf{x} = [x_1, \dots, x_9]^\mathsf{T}$ again denote a random vector which may represent the pixels of a 3×3 image. The assumption that a pixel is conditionally independent of all others given its direct horizontal and vertical neighbors directly gives rise to the MRF shown in the left part of Fig. 2.1(b). By the Hammersley-Clifford theorem, a distribution with these conditional independence assumptions can be written as

$$p(\mathbf{x}) = \frac{1}{Z}\varphi_1(x_1, x_2)\varphi_2(x_2, x_3)\varphi_3(x_1, x_4)\varphi_4(x_2, x_5)\varphi_5(x_3, x_6)\varphi_6(x_4, x_5)\cdot \varphi_7(x_5, x_6)\varphi_8(x_4, x_7)\varphi_9(x_5, x_8)\varphi_{10}(x_6, x_9)\varphi_{11}(x_7, x_8)\varphi_{12}(x_8, x_9) \quad (2.9)$$

By design, the "central" node x_5 is conditionally independent of all others given its 4 observed directly neighboring nodes (shaded gray in the right part of Fig. 2.1(b)):

$$p(x_5 | \{\mathbf{x}\} \backslash \{x_5\}) = p(x_5 | x_2, x_4, x_6, x_8). \quad (2.10)$$

Again, this applies regardless of the size of the image to be modeled.

2.1.2.1 *Factor graphs*

So far, we have glossed over the fact that an undirected GM can have cliques of several sizes. If this is the case, we can choose potentials to model either smaller or larger cliques, or a mixture thereof. In other words, we can choose between different *factorizations* of the probability distribution. We can make this explicit by defining

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{f \in \mathcal{F}} \varphi_f(\mathbf{x}_{(f)}), \quad (2.11)$$

where $f \in \mathcal{F}$ indicates the subset of nodes that correspond to factor $\varphi_f(\mathbf{x}_{(f)})$. We can visualize this with a *factor graph*, where factors are depicted with black squares (*cf.* Fig. 2.2). A factor graph is a *bipartite* graph, since there are only edges between nodes and factors, which indicate their relationships. The advantage of a factor graph over an MRF is that it makes the factorization explicit. For example, the GM shown in Fig. 2.2(a) contains cliques of up to four nodes and both factor graphs in Fig. 2.2(b,c) are equivalent to it (with respect to their conditional independences); however, Fig. 2.2(b) only uses factors (cliques) of size 2, whereas Fig. 2.2(c) is based on factors of size 4.

While the chosen size of the factors (cliques) does not change the conditional independences of the resulting distribution, it does affect its "modeling power". More complex dependencies between variables can be modeled if we use *maximal* cliques, *i. e.* cliques where no other node can be added.

2.2 MRF IMAGE MODELS

While there are MRFs that are equivalent to Bayesian networks (and vice versa), directed and undirected GMs can encode different independence assumptions and are thus not equivalent in general [*cf.* Koller and Friedman, 2009, § 4.5]. However, both kinds of GMs can be used to model images (*cf.* Fig. 2.1), *e. g.* for regularization in image restoration. Although there are some notable exceptions [*e. g.*, Domke et al., 2008; Theis et al., 2012], it is much less common to use directed GMs for this purpose. One reason may be that it seems unnatural to model parent-child relationships between pixels, although the direction of edges in the GM does not necessarily imply a (causal) order of pixels. Furthermore, the Markov blanket of a pixel will not be "symmetric" with respect to its local neighborhood (*cf.* Fig. 2.1(a)). MRFs do not suffer from these problems, as the Markov blanket of a pixel can directly be chosen through undirected connections to other (neighboring) pixels. Additionally, MRFs do not impose an artificial order on pixels, since all connections between pixels are undirected. These may be reasons why MRFs are vastly more popular image models, as compared to Bayesian networks. All image models in this dissertation are undirected GMs, whose fundamentals we discuss next.

2.2.1 *Pairwise MRFs and image statistics*

Pairwise MRFs are arguably some of the simplest sensible image models, especially when a pixel is only connected to its direct horizontal and vertical neighbors; these are most widely used in practice. The example discussed in Section 2.1.2 is actually such a pairwise MRF, where Fig. 2.1(b) shows the GM for an image of height and width 3 (9 pixels); Fig. 2.4(a) depicts the corresponding factor graph, although in this case there is actually no ambiguity w.r.t. the factorization as there are only cliques (factors) of size 2.

In natural images, we typically assume that the dependencies between neighboring pixels do not depend on their location. Hence, we are going to use the same potential function for all cliques $c \in \mathcal{C}$, *i.e.* $\varphi_c = \varphi$, implying no dependence on c. We define the probability distribution as

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \varphi(\mathbf{x}_{(c)}), \tag{2.12}$$

where $\mathcal{C}$ denotes all pairs of neighboring pixels (horizontal and vertical). Imposing such domain knowledge gives rise to a *translation-invariant* image model, since the probability (density) of an image under the model would not change if the image contents are shifted (at least for images of infinite size). Furthermore, the overall number of model parameters will be greatly reduced since all potentials *share parameters*.

We do not distinguish between horizontal and vertical potentials to ease exposition. Furthermore, statistics of horizontal and vertical cliques are often very similar in natural images.

As mentioned in Chapter 1, we model (the brightness of) each pixel with a real number, hence $\mathbf{x} \in \mathbb{R}^D$ for an image with D pixels. Next, we need to specify the potential function φ to model the dependencies between neighboring pixels. To that end, we exploit the smoothness properties of natural images [Ruderman, 1994], *i.e.* that the *difference* of neighboring pixels is mostly very small (*cf.* Fig. 2.3(a)). Hence, we want $\varphi(\mathbf{x}_{(c)})$ to return a larger value if the pixels in clique c are similar, as compared to the value that is returned when the pixels are dissimilar. Let us first define

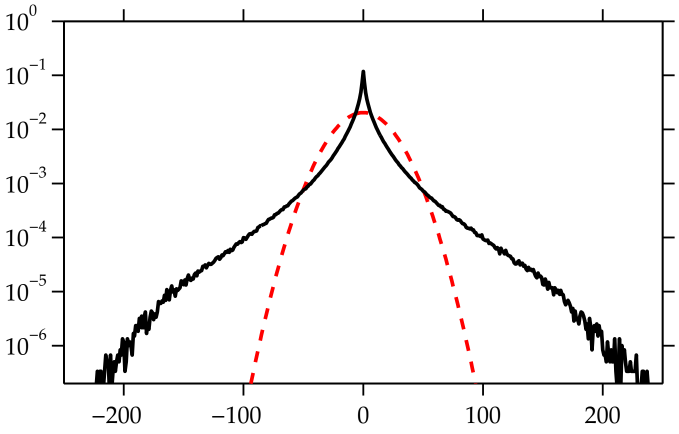
$$\varphi(\mathbf{x}_{(c)}) = \exp(-\rho(\mathbf{f}^\mathsf{T}\mathbf{x}_{(c)})), \tag{2.13}$$

where $\mathbf{f}^\mathsf{T}\mathbf{x}_{(c)}$ denotes the (brightness) difference of the two pixels belonging to clique c; hence, $\mathbf{f} = [-1, 1]^\mathsf{T}$ is a derivative *filter* and $\mathbf{x}_{(c)} \in \mathbb{R}^2$ a vector of the two pixels. To encourage smooth images, we need to choose a *penalty* function $\rho : \mathbb{R} \to \mathbb{R}$ that returns its minimum when neighboring pixels are identical, *i.e.* $\min_u \rho(u) = \rho(0)$. We also want ρ to be an *even* function that returns the same value regardless of the sign of $u = \mathbf{f}^\mathsf{T}\mathbf{x}_{(c)}$, *i.e.* $\rho(u) = \rho(-u)$; this is because we deem transitions in images from brighter to darker areas and vice versa equally likely.

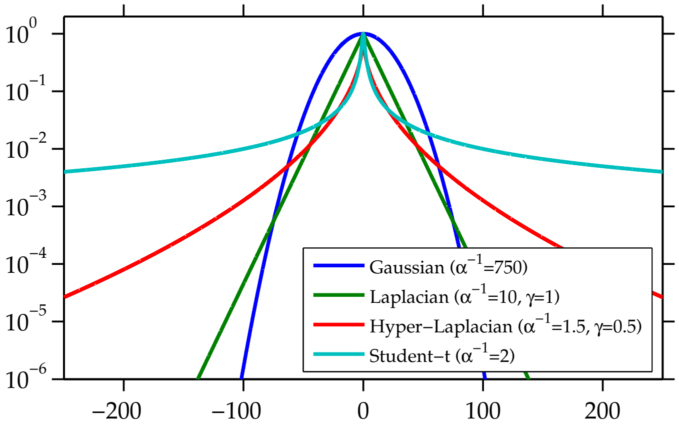
An obvious choice that satisfies these criteria is the quadratic penalty $\rho(u) = \alpha u^2$ with parameter α, which gives rise to a Gaussian potential function, since $\exp(-\rho(u)) \propto \mathcal{N}(u; 0, (2\alpha)^{-1})$. However, we will explain below why this is not a good choice.

While natural images are mostly very smooth, this does not hold at edges due to object discontinuities, strong textures, *etc.*; these "smoothness outliers" are clearly visible as *heavy tails* in the empirical distribution of neighboring pixel differences in natural images (Fig. 2.3(a), also called *marginal derivative statistics*). Roughly speaking, the probability (density) of larger (outlier) values decreases much slower as compared to a Gaussian distribution (shown in red in Fig. 2.3(a)). Using a Gaussian potential (quadratic penalty), large pixel differences will be penalized too strongly, such that images with (strong) edges will be assigned (very) low probability (density) from the MRF distribution.

Heavy tails and "peakedness" of a distribution can be quantified with the concept of (excess) kurtosis. *We do not attempt this here.*



(a) Derivative marginals of natural images



(b) Comparison of potential functions

Figure 2.3: *(a)* Empirical distribution of (horizontal and vertical) derivative filters applied to a database of natural images (black, solid); a Gaussian distribution with the same moments is shown for comparison (red, dashed). *(b)* Comparison of a Gaussian potential with other commonly-used robust potential functions.

MRFs that use heavy-tailed potential functions are also called sparse *image priors.*

As a result, many alternative potential functions with different (heavy-tailed) shapes have been proposed in the literature, which are also called *robust* or *edge-preserving* potentials; see Black and Rangarajan [1996, Appendix A] for an overview. Examples include the *Student-t* (Lorentzian) potential with $\rho(u) = \alpha \log(1 + u^2)$, which has often been used in MRFs where parameter α (among others) is learned from example data [Roth and Black, 2009; Samuel and Tappen, 2009; Chen et al., 2013]. The family of *(hyper-)Laplacian* potentials with $\rho(u) = \alpha|u|^\gamma, 0 < \gamma \leq 1$ has been also been popular [*e.g.*, Levin et al., 2007; Wang et al., 2008; Krishnan and Fergus, 2009]. A comparison of potentials is shown in Fig. 2.3(b).

Schmidt et al. [2010] have investigated the question what the "correct" potential function for a pairwise MRF image prior should be. They did this by simulating synthetic images (*i.e.* drawing random vectors) from the MRF probability distribution, which then are compared to real natural images. With this approach, they could show which potential function has to be used such that the marginal derivative statistics of simulated images from the MRF match those of natural images.

While a pairwise MRF as presented here is quite an effective image model, it is limited in the pixel dependencies that can be modeled. Recall that we connect a pixel only to its direct horizontal and vertical neighbors in the GM, which directly corresponds to its Markov blanket. To model more complex pixel dependencies, we have to enlarge a pixel's Markov blanket by introducing more edges in the GM (*cf*. Fig. 2.2(a)). Furthermore, we can define the MRF based on its maximal cliques and define potential functions to directly model larger groups of neighboring pixels (*cf*. Fig. 2.2(c)). This is what we are going to explore next.

2.2.2 *Fields of experts*

As mentioned above, to design a more powerful MRF image model, we connect a pixel to more of its neighbors. Let us assume for now that we connect a pixel to its 8 closest neighbors, as shown in Fig. 2.2(a). Furthermore, we want to model the maximal cliques of the MRF, which are now of size 4, as depicted in the factor graph of Fig. 2.2(c). As a result, we need to define a potential on 4 variables, which is more difficult as compared to the pairwise MRF. Roth and Black [2009] proposed to define the value of the potential function via a *product of experts* [Hinton, 2002], which combines the scores from several so-called *experts* through multiplication. Each expert typically models only a lower-dimensional subspace of the data. By taking the product of these experts, a low score from even one of them will result in an overall low value of the potential function. The resulting MRF is called *Field of Experts* (FoE), since the potentials are specified through products of experts.

In particular, Roth and Black [2009] defined the potentials as

$$\varphi(\mathbf{x}_{(c)}) = \prod_{i=1}^{N} \exp(-\rho_i(\mathbf{f}_i^{\mathsf{T}} \mathbf{x}_{(c)})), \tag{2.14}$$

where each of N experts $\exp(-\rho_i(\mathbf{f}_i^{\mathsf{T}} \mathbf{x}_{(c)}))$ models a *filter response* $\mathbf{f}_i^{\mathsf{T}} \mathbf{x}_{(c)}$ (*i.e.*, linear combination of clique pixels) via penalty function ρ_i. The entries of each filter $\mathbf{f}_i$ are constrained to sum to zero, *i.e.* $\sum_j \mathbf{f}_{ij} = 0$, which can be motivated by the fact that empirical distributions of (even random) zero-sum filter responses are also mostly smooth [*cf*. Huang, 2000], similar to that obtained with a derivative filter (Fig. 2.3(a)). Hence, the penalty functions ρ_i are also similar to those used in a pairwise MRF (Roth and Black [2009] used Student-t experts).

Each $\mathbf{f}_i$ denotes the vector representation of a 2D filter.

In a slight abuse of terminology, note that we will in later chapters refer to each expert as a potential function.

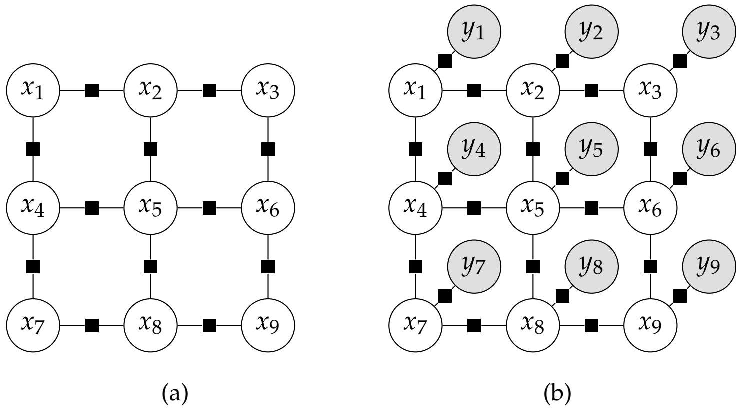


Figure 2.4: *(a)* Pairwise MRF from Fig. 2.1(b) shown as factor graph. *(b)* Factor graph of a posterior distribution obtained from a pairwise MRF image prior *(a)* and an image denoising likelihood (Eq. 1.2).

FoEs are typically used with "square" cliques (and thus filters) of odd sizes m, *i. e.* in all overlapping image patches of size $m \times m$, the central pixel is connected via an edge in the GM to all other pixels of that patch. In the literature [*e. g.*, Roth and Black, 2009; Schmidt et al., 2010; Chen et al., 2013; Schmidt and Roth, 2014], the most common FoE configurations are: 8 experts with 3×3 filters, 24 experts with 5×5 filters, and 48 experts with 7×7 filters.

Obviously, the FoE can model more complex pixel dependencies as the filter/clique sizes grow and the number of experts increases. However, the number of model parameters also goes up. Although FoEs have proven to be much better image models compared to pairwise MRFs [Roth and Black, 2009; Schmidt et al., 2010], they have at least dozens of parameters which are difficult to choose manually. Therefore, parameter learning from example data is necessary to obtain good models (*cf.* Sections 2.3.2 and 2.4.1).

Overall, the FoE probability distribution is defined as

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \prod_{i=1}^{N} \exp(-\rho_i(\mathbf{f}_i^\mathsf{T} \mathbf{x}_{(c)})), \tag{2.15}$$

which subsumes many MRF models that have been proposed in the literature [*e. g.*, Geman and Reynolds, 1992; Tappen, 2007; Krishnan and Fergus, 2009]. In particular, the pairwise MRF from the previous section can be retained as a special case of the FoE.

2.2.3 *Posterior distribution*

So far, we used GMs to encode prior knowledge about images, independent of a particular application. As briefly discussed in Chapter 1, such an image prior $p(\mathbf{x})$ is used for regularization by combining it with an application-specific likelihood $p(\mathbf{y}|\mathbf{x})$ to obtain a posterior distribution $p(\mathbf{x}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}) \cdot p(\mathbf{x})$ via Bayes' rule.

The likelihood and posterior can also be understood as GMs. In particular, the deblurring likelihood from Eq. (1.4) can be written as

$$p(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{D'} \varphi'(y_i, \mathbf{x}) \quad \text{with} \quad \varphi'(y_i, \mathbf{x}) = \mathcal{N}\Big(y_i; \sum_{j=1}^{D} K_{ij} x_j, \sigma^2\Big), \tag{2.16}$$

where cliques contain nodes from the latent image $\mathbf{x} \in \mathbb{R}^D$ and the observed image $\mathbf{y} \in \mathbb{R}^{D'}$. Also recall that the image denoising likelihood from Eq. (1.2) is retained as a special case where $K_{ij} = 0$ for all $i \neq j$ and $K_{ii} = 1$. The posterior can thus be written as

$$p(\mathbf{x}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}) \cdot p(\mathbf{x}) \propto \prod_{i=1}^{D'} \varphi'(y_i, \mathbf{x}) \cdot \prod_{c \in \mathcal{C}} \varphi(\mathbf{x}_{(c)}). \tag{2.17}$$

Note that if the matrix $\mathbf{K}$ contains many non-zero entries, the likelihood cliques introduce many edges between the pixels x_i of the posterior GM. This is not the case for image denoising, however, where no additional edges between the pixels of the latent image are introduced, as shown in Fig. 2.4(b) for a pairwise MRF prior; the factors introduced by the likelihood that connect x_i and y_i are also called *unary potentials*.

2.2.3.1 *Conditional random fields*

Since only the posterior distribution is required in a particular application, for instance to predict a deblurred image, a *conditional random field* (CRF) [Lafferty et al., 2001] directly models the posterior as

$$p(\mathbf{x}|\mathbf{y}) = \frac{1}{Z} \prod_{c \in \mathcal{C}''} \varphi''(\mathbf{x}_{(c)}, \mathbf{y}) \tag{2.18}$$

without separately specifying likelihood and prior, as in a generative approach. This has the advantage that clique potentials φ'' can be used that have access to the entire observed image $\mathbf{y}$, which is always assumed to be known. Furthermore, the posterior distribution might be easier to model than prior and likelihood. However, such a discriminative approach is no longer application-independent. Hence, the specific form of clique potentials φ'' is different depending on the particular application.

2.3 PROBABILISTIC INFERENCE AND LEARNING

Whether we follow a generative approach and model prior and likelihood separately, or use a discriminative approach, we end up with a posterior distribution over the restored image. However, we typically need to predict a single restored image, hence have to make a decision based on the posterior distribution. Although the following discussion is in the context of image restoration, most principles apply in general.

2.3.1 *Bayesian point estimation*

This assumption can be problematic, cf. *Section 2.5.*

Under the assumption that the posterior is accurate, Bayesian *decision theory* [*e. g.*, Berger, 1985] tells us that the optimal prediction is obtained by choosing the image $\hat{\mathbf{x}}$ that minimizes the expected value of a *loss function* Δ (the *Bayesian expected loss*) over the posterior distribution:

$$\hat{\mathbf{x}} = \arg\min_{\tilde{\mathbf{x}}} \mathbb{E}\left[\Delta(\tilde{\mathbf{x}}, \mathbf{x}) | \mathbf{y}\right] = \arg\min_{\tilde{\mathbf{x}}} \int \Delta(\tilde{\mathbf{x}}, \mathbf{x}) p(\mathbf{x}|\mathbf{y})\, d\mathbf{x}. \tag{2.19}$$

Making a prediction like this is also called the *Bayes estimator* for the chosen loss function $\Delta(\tilde{\mathbf{x}}, \mathbf{x})$, which assigns a loss (*i. e.*, cost) to prediction $\tilde{\mathbf{x}}$ if the correct value was $\mathbf{x}$. This loss function should be based on the criterion that is used to evaluate the prediction for the particular application.

$\mathbb{I}$ *denotes an indicator function.*

Let us assume that we have chosen the *0-1 loss* function

$$\Delta(\tilde{\mathbf{x}}, \mathbf{x}) = \mathbb{I}[\tilde{\mathbf{x}} \neq \mathbf{x}] = \begin{cases} 1 & \text{if } \tilde{\mathbf{x}} \notin \{\mathbf{x}\} \\ 0 & \text{if } \tilde{\mathbf{x}} \in \{\mathbf{x}\} \end{cases} \tag{2.20}$$

that assigns no cost to the correct prediction, and returns a cost of 1 otherwise. The Bayes estimator for the 0-1 loss is the well-known *maximum a-posteriori* (MAP) estimate, which corresponds to the most probable state of the posterior:

$$\begin{aligned} \hat{\mathbf{x}} &= \arg\min_{\tilde{\mathbf{x}}} \int \mathbb{I}[\tilde{\mathbf{x}} \neq \mathbf{x}] p(\mathbf{x}|\mathbf{y})\, d\mathbf{x} \\ &= \arg\min_{\tilde{\mathbf{x}}} 1 - p(\tilde{\mathbf{x}}|\mathbf{y}) \\ &= \arg\max_{\tilde{\mathbf{x}}} p(\tilde{\mathbf{x}}|\mathbf{y}). \end{aligned} \tag{2.21}$$

Although the 0-1 loss is not particularly suitable in the context of image restoration (and many other applications), MAP estimation is commonly done since it can be carried out via a (comparatively) simple optimization problem that does not even require the posterior distribution to be normalized.

However, it is often intractable to minimize the expected loss in Eq. (2.19) for general posterior distributions and loss functions. The problem typically lies in computing the integral over all possible states $\mathbf{x} \in \mathbb{R}^D$, which in our case are all images with D pixels. Even if every pixel was modeled as a binary variable, integration would correspond to summation over 2^D possible discrete states. Unfortunately, such integrations (or summations) over an exponential state space can often not be simplified and thus are impractical.

Since it is often the case that a loss function $\Delta(\tilde{\mathbf{x}}, \mathbf{x}) = \sum_{i=1}^{D} \Delta_i(\tilde{x}_i, x_i)$ decomposes as a sum over individual variables, one approach to make Eq. (2.19) tractable is to make a *mean field* approximation [*e. g.*, Geiger and Girosi, 1991; Chantas et al., 2008; Schelten and Roth, 2012] of the posterior with a distribution q that factorizes over individual (or a small number of) variables, *e. g.*:

$$p(\mathbf{x}|\mathbf{y}) \approx q(\mathbf{x}|\mathbf{y}) = \prod_{i=1}^{D} q(x_i|\mathbf{y}). \tag{2.22}$$

Using q instead of p and a decomposable loss Δ, we can approximate $\hat{\mathbf{x}}$ in Eq. (2.19) as

$$\begin{aligned} \hat{\mathbf{x}} &\approx \arg\min_{\tilde{\mathbf{x}}} \int \sum_{i=1}^{D} \Delta_i(\tilde{x}_i, x_i) q(\mathbf{x}|\mathbf{y})\, d\mathbf{x} \\ &= \arg\min_{\tilde{\mathbf{x}}} \sum_{i=1}^{D} \int \cdots \int \Delta_i(\tilde{x}_i, x_i) \prod_{j=1}^{D} q(x_j|\mathbf{y})\, dx_1 \dots dx_D \\ &= \arg\min_{\tilde{\mathbf{x}}} \sum_{i=1}^{D} \int \Delta_i(\tilde{x}_i, x_i) q(x_i|\mathbf{y})\, dx_i. \end{aligned} \tag{2.23}$$

This is much easier to compute, since the D-dimensional integral has been replaced with a sum of D 1-dimensional integrals.

In this dissertation, though, we make Eq. (2.19) accessible by directly approximating the D-dimensional integral. Since the expected value of a function f

$$\mathbb{E}[f(\mathbf{x})|\mathbf{y}] = \int f(\mathbf{x}) p(\mathbf{x}|\mathbf{y})\, d\mathbf{x} \approx \frac{1}{T} \sum_{t=1}^{T} f\left(\mathbf{x}^{(t)}\right) \tag{2.24}$$

can always be approximated with a set of samples $\{\mathbf{x}^{(t)}\}_{t=1}^{T} \sim p(\mathbf{x}|\mathbf{y})$ from the posterior [*cf.* Bishop, 2006, § 11], Eq. (2.19) is approximated as:

$$\hat{\mathbf{x}} \approx \arg\min_{\tilde{\mathbf{x}}} \frac{1}{T} \sum_{t=1}^{T} \Delta(\tilde{\mathbf{x}}, \mathbf{x}^{(t)}). \tag{2.25}$$

The quality of the approximation depends on the number of (independent) samples.

It is relatively easy to draw samples from some distributions, such as a (multivariate) Gaussian (*cf.* Section 3.5.3)

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{Z} \exp\left(-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^\mathsf{T} \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right) \tag{2.26}$$

with tractable normalization constant

$$Z = \int \exp\left(-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^\mathsf{T} \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right) d\mathbf{x} = (2\pi)^{-D/2} |\boldsymbol{\Sigma}|^{-1/2}. \tag{2.27}$$

Unfortunately, it is generally difficult to draw samples from MRFs, in part due to normalization constants Z (*cf.* Eq. 2.7) that often do not admit tractable integration. Hence, we will now discuss the *Gibbs sampler* [Geman and Geman, 1984], which can be used to draw samples from MRFs and other, unnormalized distributions.

2.3.1.1 *Gibbs sampling*

Gibbs sampling is an iterative sampling procedure, where we first initialize $\mathbf{x}^{(0)}$ with an arbitrary value (from the domain of $\mathbf{x}$). At every step $t = 1, \ldots, T$ of the algorithm, we derive from the previous state $\mathbf{x}^{(t-1)}$ an updated state $\mathbf{x}^{(t)}$ by sampling a new value for each variable x_i ($i = 1, \ldots, D$) via its conditional distribution

$$x_i^{(t)} \sim p(x_i | x_1^{(t)}, \ldots, x_{i-1}^{(t)}, x_{i+1}^{(t-1)}, \ldots, x_D^{(t-1)}, \mathbf{y}) \tag{2.28}$$

while holding all other variables fixed at their most recent value. The update order of variables can be freely chosen (also random).

While directly sampling from the D-dimensional posterior $p(\mathbf{x}|\mathbf{y})$ is generally difficult, sampling from the 1-dimensional conditionals is often much simpler, especially because they depend only on the variables of their Markov blanket (Eq. 2.6) in an MRF.

The obtained sequence $\mathbf{x}^{(0)}, \ldots, \mathbf{x}^{(T)}$ forms a *Markov chain*, where every state $\mathbf{x}^{(t)}$ was produced only from the preceding $\mathbf{x}^{(t-1)}$. If we assume that all conditionals $p(x_i | \{\mathbf{x}\} \setminus \{x_i\}) > 0$ everywhere, then it can be shown [*e.g.*, Bishop, 2006, § 11.2] that such a Markov chain obtained via Gibbs sampling convergences to the desired distribution $p(\mathbf{x}|\mathbf{y})$. This means after a number of B steps (called the *burn-in* phase), the Markov chain will have "forgotten" its initial state $\mathbf{x}^{(0)}$ and converged to the desired distribution; the sequence $\mathbf{x}^{(B+1)}, \ldots, \mathbf{x}^{(T)}$ then denotes (dependent) samples from $p(\mathbf{x}|\mathbf{y})$. Gibbs sampling belongs to the class of *Markov chain Monte Carlo* (MCMC) methods [*cf.* Andrieu et al., 2003], and can in particular be shown to be a special case of the *Metropolis-Hastings* [Metropolis et al., 1953; Hastings, 1970] algorithm.

The Markov chain is said to suffer from poor *mixing* if neighboring states of the chain are strongly dependent. Unfortunately, this can frequently happen in MRF image models when neighboring pixels are strongly dependent due to the particular choice of potential functions; hence, updating a single pixel by re-sampling from its conditional distribution will often result in little change. A way to improve on that is to use *block* Gibbs sampling by updating multiple variables (pixels) jointly by re-sampling from their (joint) conditional distribution. However, jointly re-sampling several pixels typically does not scale beyond a few variables.

For the MRF image models in this dissertation, block Gibbs sampling can be made much more practical by first introducing auxiliary variables $\mathbf{z}$ to define an augmented distribution $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$, such that the desired distribution $p(\mathbf{x}|\mathbf{y}) = \int p(\mathbf{x}, \mathbf{z}|\mathbf{y})\, d\mathbf{z}$ is retained via marginalization [*cf.* Gelman et al., 2004, § 11.8]. The auxiliary vector $\mathbf{z}$ is introduced in such a way that the conditional distributions $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$ and $p(\mathbf{z}|\mathbf{x}, \mathbf{y})$ facilitate good mixing and are both easy to sample from. Hence, block Gibbs sampling in this case alternates between sampling from these two conditionals to produce the sequence

$\mathbf{x}^{(0)}, \mathbf{z}^{(1)}, \mathbf{x}^{(1)}, \mathbf{z}^{(2)}, \ldots, \mathbf{x}^{(T)}$, which eventually yields samples from the augmented distribution $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$. In the end, we can discard the samples for $\mathbf{z}$ and simply keep those for $\mathbf{x}$, which are distributed according to $p(\mathbf{x}|\mathbf{y})$. The specific form of our auxiliary variable block Gibbs sampler will be explained in Chapter 3.

Although we discussed Gibbs sampling in the context of posterior inference, it can equally be applied to draw samples from the MRF prior distribution $p(\mathbf{x})$, which will be relevant for (unsupervised) learning in the next section.

2.3.2 *Maximum likelihood learning*

We have discussed probabilistic posterior inference with MRF-based image models. However, although we explained the general architecture of these models, we have not yet explained how to specify them exactly. Whether we directly want to model the posterior $p(\mathbf{x}|\mathbf{y})$ with a CRF or use a generative MRF prior $p(\mathbf{x})$, the respective random field models actually denote a family of distributions, which are only fully specified given a set of model parameters $\mathbf{\Theta}$. To make the dependence on model parameters explicit, we will write $p(\mathbf{x}) \equiv p(\mathbf{x}; \mathbf{\Theta})$ and $p(\mathbf{x}|\mathbf{y}) \equiv p(\mathbf{x}|\mathbf{y}; \mathbf{\Theta})$ in the remainder of this chapter.

Let us consider the FoE image prior from Eq. (2.15) as a specific example, which we can write without loss of generality as

$$p(\mathbf{x}; \mathbf{\Theta}) = \frac{1}{Z(\mathbf{\Theta})} \exp(-E(\mathbf{x}; \mathbf{\Theta})) \quad \text{with energy} \tag{2.29}$$

$$E(\mathbf{x}; \mathbf{\Theta}) = \sum_{c \in \mathcal{C}} \sum_{i=1}^{N} \rho(\mathbf{f}_i^\mathsf{T} \mathbf{x}_{(c)}; \boldsymbol{\alpha}_i), \tag{2.30}$$

assuming that all penalty functions can be specified as $\rho_i(u) = \rho(u; \boldsymbol{\alpha}_i)$. The FoE is fully specified given all linear filters and their associated penalty functions, hence the model parameters are $\mathbf{\Theta} = \{\mathbf{f}_i, \boldsymbol{\alpha}_i\}_{i=1}^N$. Note that the intractable normalization constant

$$Z(\mathbf{\Theta}) = \int \exp(-E(\mathbf{x}; \mathbf{\Theta}))\, d\mathbf{x} \tag{2.31}$$

depends on all model parameters, which will make learning complicated as we shall see shortly. We could similarly consider a CRF model

$$p(\mathbf{x}|\mathbf{y}; \mathbf{\Theta}) = \frac{1}{Z(\mathbf{\Theta}, \mathbf{y})} \exp(-E(\mathbf{x}|\mathbf{y}; \mathbf{\Theta})) \tag{2.32}$$

whose normalization constant also depends on observed image $\mathbf{y}$.

We will now discuss how the model parameters $\mathbf{\Theta}$ of a generic image prior $p(\mathbf{x}; \mathbf{\Theta}) = \frac{1}{Z(\mathbf{\Theta})} \exp(-E(\mathbf{x}; \mathbf{\Theta}))$ can (approximately) be learned from a given set of M training examples $\mathfrak{D} = \{\mathbf{x}^{(i)}\}_{i=1}^M$ with the well-known method of *maximum likelihood*. However, our exposition will equally apply to learning the parameters of a CRF posterior

$p(\mathbf{x}|\mathbf{y};\boldsymbol{\Theta})$, with the difference that in this case the training data must consist of input-output pairs $\mathfrak{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{M}$. Note that learning a prior is also called *unsupervised* learning, whereas posterior training is called *supervised* learning.

Let us assume that the training data $\mathfrak{D}$ consists of i.i.d. samples from the distribution that we want to approximate with our model distribution $p(\mathbf{x};\boldsymbol{\Theta})$ from Eq. (2.29). To that end, in maximum likelihood learning we want to find a single set of parameters $\hat{\boldsymbol{\Theta}}$ that maximizes the probability of the training data under the model distribution. Mathematically, we can formalize this is as the argument $\hat{\boldsymbol{\Theta}} = \arg\max_{\boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\Theta};\mathfrak{D})$ that maximizes the log-likelihood function

$$\mathcal{L}(\boldsymbol{\Theta};\mathfrak{D}) = \log \prod_{i=1}^{M} p(\mathbf{x}^{(i)};\boldsymbol{\Theta}) = -M \log Z(\boldsymbol{\Theta}) - \sum_{i=1}^{M} E(\mathbf{x}^{(i)};\boldsymbol{\Theta}). \tag{2.33}$$

Unfortunately, we cannot even evaluate the log-likelihood due to the intractable normalization constant $Z(\boldsymbol{\Theta})$.

Before we address this, let use first define

$$\langle f(\mathbf{x}) \rangle_{\mathfrak{B}} = \frac{1}{|\mathfrak{B}|} \sum_{\mathbf{x} \in \mathfrak{B}} f(\mathbf{x}) \tag{2.34}$$

to denote the expected value of a function f over the empirical distribution $q(\mathbf{u};\mathfrak{B}) = \frac{1}{|\mathfrak{B}|} \sum_{\mathbf{x} \in \mathfrak{B}} \mathbb{I}[\mathbf{x} = \mathbf{u}]$ defined by dataset $\mathfrak{B}$. Using this definition, we can write Eq. (2.33) as

$$\mathcal{L}(\boldsymbol{\Theta};\mathfrak{D}) = -M \log Z(\boldsymbol{\Theta}) - M \left\langle E(\mathbf{x};\boldsymbol{\Theta}) \right\rangle_{\mathfrak{D}}. \tag{2.35}$$

Let us assume that the model energy is differentiable w.r.t. the model parameters $\boldsymbol{\Theta}$. Then, although we cannot evaluate the log-likelihood, we are able to approximate its derivative w.r.t. $\boldsymbol{\Theta}$ as

$$\begin{aligned}
\frac{\partial \mathcal{L}(\boldsymbol{\Theta};\mathfrak{D})}{\partial \boldsymbol{\Theta}} &= -M \frac{\partial \log Z(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} - \sum_{i=1}^{M} \frac{\partial E(\mathbf{x}^{(i)};\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \\
&= -M \frac{1}{Z(\boldsymbol{\Theta})} \frac{\partial \int \exp(-E(\mathbf{x};\boldsymbol{\Theta}))\, d\mathbf{x}}{\partial \boldsymbol{\Theta}} - M \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \right\rangle_{\mathfrak{D}} \\
&= M \int \frac{\partial E(\mathbf{x};\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} p(\mathbf{x};\boldsymbol{\Theta})\, d\mathbf{x} - M \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \right\rangle_{\mathfrak{D}} \\
&\approx M \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \right\rangle_{\mathfrak{S}} - M \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \right\rangle_{\mathfrak{D}},
\end{aligned} \tag{2.36}$$

where $\mathfrak{S} = \{\mathbf{x}^{(i)}\}_{i=1}^{K} \sim p(\mathbf{x};\boldsymbol{\Theta})$ denotes a set of samples drawn from the model distribution. As a consequence, we can use gradient ascent to find the parameters $\hat{\boldsymbol{\Theta}}$ that (approximately) maximize the log-likelihood. Unfortunately, this is computationally rather expensive because after every step of gradient ascent the model parameters change and we have to again draw samples from the model distribution.

We assume that the model samples are obtained via an iterative Markov chain Monte Carlo method, such as the Gibbs sampler explained in the previous section. If we initialize a separate Markov chain for each element of a dataset $\mathfrak{B}$, then we can denote by $\mathfrak{B}^t$ the state of all chains after t steps. This definition allows us to write Eq. (2.36) as

$$\frac{\partial \mathcal{L}(\boldsymbol{\Theta};\mathfrak{D})}{\partial \boldsymbol{\Theta}} \approx M \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \right\rangle_{\mathfrak{D}^{\infty}} - M \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \right\rangle_{\mathfrak{D}^{0}}, \tag{2.37}$$

since each Markov chain will have forgotten its initialization if run for long enough, *i. e.* $\mathfrak{S} \equiv \mathfrak{D}^{\infty}$. To speed this up, Hinton [2002] proposed an approximation by running each Markov chain only for a small number of t steps:

$$\frac{\partial \mathcal{L}(\boldsymbol{\Theta};\mathfrak{D})}{\partial \boldsymbol{\Theta}} \approx M \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \right\rangle_{\mathfrak{D}^{t}} - M \left\langle \frac{\partial E(\mathbf{x};\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \right\rangle_{\mathfrak{D}^{0}}. \tag{2.38}$$

Learning with this approach is known as *contrastive divergence* (CD) and typically gives reasonable results even for $t = 1$. It can be argued that CD has revived learning of unnormalized models, such as MRFs. Carreira-Perpiñán and Hinton [2005] show that although CD is a biased estimator, it often yields estimates quite similar to maximum likelihood.

Instead of always initializing the Markov chains with the training data $\mathfrak{D}$ at every step of gradient ascent, we can for each chain also remember its state after advancing it. When doing this, after many iterations of gradient ascent, the states of the Markov chains will intuitively be quite close to unbiased samples from the current model distribution, because the model parameters $\boldsymbol{\Theta}$ change only slightly after each step of gradient ascent. This approach has been popularized by Tieleman [2008] under the name *persistent* contrastive divergence (PCD), although it has been proposed much earlier by Younes [1989]. In practice, PCD can work even better than CD [*e. g.*, Tieleman, 2008; Gao and Roth, 2012].

Due to the intractable normalization constant, maximum likelihood learning of MRFs is unfortunately quite involved. Not only is it necessary to approximate expectations via sampling (Eq. 2.36), but even sampling itself needs to be approximated (CD, PCD) to carry out learning in a reasonable amount of time. Furthermore, we cannot directly monitor progress during learning, since the log-likelihood function itself cannot be evaluated. These are reasons why alternatives to maximum likelihood have recently been proposed, such as *score matching* [Hyvärinen, 2005], *noise contrastive* estimation [Gutmann and Hyvärinen, 2012], and *minimum probability flow* learning [Sohl-Dickstein et al., 2011].

Also note that parameter learning in Bayesian network (image) models is typically much simpler [Domke et al., 2008] because directed GMs are normalized by definition (*cf.* Eq. 2.2).

We have so far presented a probabilistic approach, where we first learn a model distribution (for prior or posterior) of samples from the real – but unknown – data distribution. Then, given an observation and a suitable loss function, we use our learned model distribution to make a prediction by minimizing the expected loss.

Concretely, whether we directly model the posterior $p(\mathbf{x}|\mathbf{y};\boldsymbol{\Theta})$ or obtain $p(\mathbf{x}|\mathbf{y};\boldsymbol{\Theta}) \propto p(\mathbf{y}|\mathbf{x})p(\mathbf{x};\boldsymbol{\Theta})$ via a likelihood and prior model, probabilistic inference obtains $\hat{\mathbf{x}} = h(\mathbf{y};\boldsymbol{\Theta},\Delta)$ by minimizing the expected loss via *prediction function*

$$h(\mathbf{y};\boldsymbol{\Theta},\Delta) = \arg\min_{\tilde{\mathbf{x}}} \int \Delta(\tilde{\mathbf{x}},\mathbf{x})\, p(\mathbf{x}|\mathbf{y};\boldsymbol{\Theta})\, d\mathbf{x}, \tag{2.39}$$

which is parameterized by model (distribution) parameters $\boldsymbol{\Theta}$ and loss function Δ; note that $\boldsymbol{\Theta}$ and Δ are chosen independently of each other. Unfortunately, estimating $\boldsymbol{\Theta}$ for a model distribution and subsequently computing $h(\mathbf{y};\boldsymbol{\Theta},\Delta)$ is hard, because probabilistic modeling necessitates computing expectations over model distributions, which we have seen is generally difficult for suitable MRF-based image priors and posteriors (including CRFs).

However, viewed as a black box from the outside, the prediction function in Eq. (2.39) looks like a *regression* function that takes $\mathbf{y}$ as input and returns $\hat{\mathbf{x}}$. Hence, an alternative non-probabilistic approach is to directly learn a prediction function $h_\Delta(\mathbf{y};\boldsymbol{\Theta})$ via $\boldsymbol{\Theta}$ by treating this as a regression problem of multiple input and output variables [*e.g.*, Samuel and Tappen, 2009; Chen et al., 2013]. Note that $h_\Delta(\mathbf{y};\boldsymbol{\Theta})$ is a discriminative model, which is specific to a particular application *and* loss function.

Note that $h_\Delta(\mathbf{y};\boldsymbol{\Theta})$ does not have to be defined as in Eq. (2.40).

Most commonly [*e.g.*, Samuel and Tappen, 2009; Pletscher et al., 2011] – and also assumed here – such a prediction function

$$h_\Delta(\mathbf{y};\boldsymbol{\Theta}) = \arg\min_{\mathbf{x}} E(\mathbf{x}|\mathbf{y};\boldsymbol{\Theta}) \tag{2.40}$$

is defined via minimization of an energy $E(\mathbf{x}|\mathbf{y};\boldsymbol{\Theta})$ associated to CRF $p(\mathbf{x}|\mathbf{y};\boldsymbol{\Theta}) \propto \exp(-E(\mathbf{x}|\mathbf{y};\boldsymbol{\Theta}))$ (Section 2.2.3.1). Although energy minimization via Eq. (2.40) corresponds to MAP estimation of $p(\mathbf{x}|\mathbf{y};\boldsymbol{\Theta})$, this analogy is only valid when $p(\mathbf{x}|\mathbf{y};\boldsymbol{\Theta})$ aims to model the posterior distribution (*e.g.*, with $\boldsymbol{\Theta}$ learned via maximum likelihood as in Section 2.3.2). Hence, it is technically not correct to refer to Eq. (2.40) as MAP estimation, since the parameters $\boldsymbol{\Theta}$ will be chosen in a different way (as discussed below).

2.4.1 *Loss-based training*

To learn the prediction function (Eq. 2.40) via parameters $\boldsymbol{\Theta}$, we assume access to i.i.d. training data (from the unknown data distribution) in the form of input-output pairs $\mathfrak{D} = \{(\mathbf{x}^{(i)},\mathbf{y}^{(i)})\}_{i=1}^{M}$. While the data assumption is the same as for training CRFs in Section 2.3.2, instead of maximizing the log-likelihood function (Eq. 2.33), we will choose $\hat{\mathbf{\Theta}} = \arg\min_{\mathbf{\Theta}} \mathcal{R}(\mathbf{\Theta}; \mathfrak{D})$ as the minimizer of the *empirical risk*

$$\mathcal{R}(\mathbf{\Theta}; \mathfrak{D}) = \sum_{i=1}^{M} \Delta\big(\mathbf{x}^{(i)}, h_\Delta(\mathbf{y}^{(i)}; \mathbf{\Theta})\big) \tag{2.41}$$

for the given loss function Δ and training data $\mathfrak{D}$. In contrast to the log-likelihood function, we can evaluate the empirical risk in Eq. (2.41) for a particular value of $\mathbf{\Theta}$ in a reasonable amount of time, hence can directly monitor progress during learning. However, this hinges on Eq. (2.40) (and loss Δ) being relatively easy to compute; often they are chosen for ease of computation.

Learning refers to solving the *nested* optimization problem

$$\arg\min_{\mathbf{\Theta}} \mathcal{R}(\mathbf{\Theta}; \mathfrak{D}) = \arg\min_{\mathbf{\Theta}} \sum_{i=1}^{M} \Delta\big(\mathbf{x}^{(i)}, \arg\min\nolimits_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}^{(i)}; \mathbf{\Theta})\big), \tag{2.42}$$

which typically cannot be computed analytically, but can be approximated with gradient-based optimization. To that end, we need to compute the derivative of Eq. (2.41) w.r.t. $\mathbf{\Theta}$, which is obtained as

$$\frac{\partial \mathcal{R}(\mathbf{\Theta}; \mathfrak{D})}{\partial \mathbf{\Theta}} = \sum_{i=1}^{M} \frac{\partial \Delta\big(\mathbf{x}^{(i)}, h_\Delta(\mathbf{y}^{(i)}; \mathbf{\Theta})\big)}{\partial h_\Delta(\mathbf{y}^{(i)}; \mathbf{\Theta})} \cdot \frac{\partial h_\Delta(\mathbf{y}^{(i)}; \mathbf{\Theta})}{\partial \mathbf{\Theta}}. \tag{2.43}$$

While differentiating the loss w.r.t. the prediction is typically very easy, the gradient

$$\frac{\partial h_\Delta(\mathbf{y}; \mathbf{\Theta})}{\partial \mathbf{\Theta}} = \frac{\partial \arg\min_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}; \mathbf{\Theta})}{\partial \mathbf{\Theta}} \tag{2.44}$$

of the prediction function w.r.t. the parameters is more challenging to obtain when $h_\Delta(\mathbf{y}; \mathbf{\Theta})$ does not have a closed-form expression. While we will choose prediction functions with closed-form expressions (Chapters 6 and 7), this is often not the case. However, Samuel and Tappen [2009] have shown how Eq. (2.44) can then be computed for CRFs by using the method of *implicit differentiation*. Furthermore, Chen et al. [2013] have generalized this based on connections to techniques from *bi-level* optimization [*cf.* Colson et al., 2007]. Additionally, one can employ *truncated* optimization of Eq. (2.40) to make this problem easier [Barbu, 2009; Domke, 2012].

While loss-specific training via empirical risk minimization can be somewhat challenging due to the need for solving nested optimization problems, it is still simpler than maximum likelihood learning. Furthermore, the form of the prediction function $h_\Delta(\mathbf{y}; \mathbf{\Theta})$ can be freely chosen. Overall, deterministic learning and inference has the big computational advantage over its probabilistic alternative that it does not require computation of expectations over model distributions (as discussed previously). However, this does not mean that it is superior to probabilistic modeling. All variants discussed in this chapter (probabilistic *vs*. deterministic, generative *vs*. discriminative) have their advantages and disadvantages, which we will briefly discuss next.

2.5 COMPARISON OF MODELING APPROACHES

In order to better understand their differences and commonalities (in the context of image restoration), we will now summarize and compare the modeling approaches introduced in this chapter.

2.5.1 *Generative*

A (probabilistic) generative approach models the joint distribution $p(\mathbf{x}, \mathbf{y}; \boldsymbol{\Theta}, \boldsymbol{\beta})$ of the observed image $\mathbf{y}$ and latent image $\mathbf{x}$, here via a likelihood model $p(\mathbf{y}|\mathbf{x}; \boldsymbol{\beta})$ of the known application-specific observation process and a prior distribution $p(\mathbf{x}; \boldsymbol{\Theta})$ of "good" images, specified by model parameters $\boldsymbol{\Theta}$. Note that we have made explicit that the likelihood $p(\mathbf{y}|\mathbf{x}) \equiv p(\mathbf{y}|\mathbf{x}; \boldsymbol{\beta})$ is determined by a few parameters $\boldsymbol{\beta}$ (*e.g.*, the noise variance in Eq. (1.2)), which are often unknown in practice. The prior distribution is independent of a particular application (such as image denoising), thus $\boldsymbol{\Theta}$ can be learned solely from a database of good example images (typically via maximum likelihood, which can be difficult, *cf*. Section 2.3.2). Employing Bayes' rule, the posterior distribution $p(\mathbf{x}|\mathbf{y}; \boldsymbol{\Theta}, \boldsymbol{\beta})$ is used to predict a single (restored) output image $\hat{\mathbf{x}}$ for the given observation $\mathbf{y}$, concretely by minimizing the expected value of a loss function over the posterior (*cf*. Section 2.3.1). If we assume the 0-1 loss, then MAP estimation is the correct decision. However, the quality metrics used in image restoration are quite different (see upcoming Section 2.6.2), such as based on the squared loss of the prediction. Unfortunately, probabilistic prediction for these suitable loss functions is computationally more difficult compared to MAP estimation. Furthermore, probabilistic inference hinges on the assumption that the model distribution is an accurate representation of the data in the real world. Hence, probabilistic approaches – generative ones in particular – are sensitive to *misspecification* [*cf*. White, 1982; Liang and Jordan, 2008; Pletscher et al., 2011], here especially by making use of simplistic model distributions. However, if the models are accurate, probabilistic approaches offer excellent results and many additional benefits, such as uncertainty estimates for predictions, and the ability to handle unobserved random variables (such as the likelihood parameters $\boldsymbol{\beta}$, *cf*. Chapter 4) in a principled way.

SUMMARY

- + *Probabilities:* can provide uncertainty estimates, allows handling of unobserved variables
- + *Versatility (Application):* prior is application-independent, can be combined with different likelihoods
- + *Versatility (Loss):* model is independent of loss function used for making predictions
- + *Data requirements:* prior can be trained from good example images alone
- – *Misspecification:* sensitive to modeling errors, model distributions often only simplistic representation of real data
- – *Modeling:* prior (or joint distribution) can be more difficult to model than posterior
- – *Learning:* learning is difficult due to the intractable normalization constant
- – *Prediction:* prediction for suitable losses is computationally demanding

2.5.2 *Discriminative*

In a (probabilistic) discriminative approach, only the posterior distribution $p(\mathbf{x}|\mathbf{y};\mathbf{\Theta})$ is modeled, which is required for predicting the restored image. Hence, this is now application-dependent, which also means that the posterior might be easier to model than the prior (or joint distribution) in a generative approach. This may also alleviate the problem of misspecification, since less of the data needs to be modeled [*cf.* Liang and Jordan, 2008]. Learning the posterior distribution requires training data of input-output pairs (also called *labeled* data), which should be representative of all input-output combinations. To enable accurate modeling, the number of required training examples is typically higher than in a generative approach. Furthermore, it might be difficult to acquire labeled data. Otherwise, learning and inference in a probabilistic discriminative approach proceeds in the same way as in the generative approach described above, thus has the same advantages and disadvantages.

SUMMARY

- + *Probabilities:* can provide uncertainty estimates, allows handling of unobserved variables
- – *Versatility (Application):* posterior is application-specific
- + *Versatility (Loss):* model is independent of loss function used for making predictions

– *Data requirements:* posterior needs input-output pairs for training, more data required as in generative approach
± *Misspecification:* less sensitive to modeling errors than generative model, but posterior distribution often still simplistic
+ *Modeling:* posterior can be simpler to model than prior (or joint distribution)
– *Learning:* learning is difficult due to the intractable normalization constant
– *Prediction:* prediction for suitable losses is computationally demanding

2.5.2.1 *Deterministic*

In a deterministic discriminative approach, only a prediction function $h_\Delta(\mathbf{y}; \mathbf{\Theta})$ is modeled that returns a restored image for a given observation $\mathbf{y}$, typically via energy minimization (*cf.* Section 2.4), which is much simpler compared to probabilistic inference. In further contrast to probabilistic modeling, the loss function Δ is already used to learn the model. Training with input-output pairs minimizes the loss of the prediction function w.r.t. the training examples, which is typically simpler than learning a probability distribution, but can also by challenging (*cf.* Section 2.4.1). The results of such a deterministic prediction function can often be better compared to probabilistic approaches, because misspecification is less of an issue [*cf.* Pletscher et al., 2011]. Given enough labeled training data, even a simplistic prediction function often generalizes well to unseen data. However, note that a deterministic prediction function is tailored to a specific application and loss function, hence is less versatile.

SUMMARY

– *Probabilities:* no probabilities, cannot handle unobserved variables or provide uncertainty estimates in a principled way
– *Versatility (Application):* prediction function is application-specific
– *Versatility (Loss):* prediction function is loss-specific
– *Data requirements:* prediction function needs input-output pairs for training, more data required as in generative approach
+ *Misspecification:* less sensitive to modeling errors than probabilistic approach
+ *Modeling:* prediction function can be simpler to model than probability distribution
± *Learning:* simpler due to absence of intractable normalization constant, but often involves nested optimization problems
+ *Prediction:* prediction much easier by design, typically via energy minimization

2.5.3 *Discussion*

As is evident from the exposition above, there is no best modeling approach to be used in all situations. Hence, which one to choose depends on the specific situation. For example, probabilistic generative approaches are very versatile, but are often problematic w.r.t. learning and inference. In contrast, deterministic discriminative approaches are very specialized, but benefit from easier learning and inference. In this dissertation, we propose novel methods in the context of (probabilistic) generative approaches (Chapters 4 and 5) and (deterministic) discriminative approaches (Chapters 6 and 7).

Given enough labeled training data, discriminative methods typically yield the best results in benchmarks for a specific (image restoration) application, such as image denoising. We will discuss next how image restoration methods are typically compared and evaluated, before we briefly survey other related work for image restoration.

2.6 IMAGE RESTORATION

2.6.1 *Denoising and deblurring*

Since we mainly address the image restoration tasks of removing noise and blur from natural images in this dissertation, we give an overview of solution approaches to these problems.

Regarding the use of graphical models, recall that only the image prior is modeled in a generative approach, which (in principle) is application-neutral and can thus be applied to many problems when combined with a suitable likelihood model. Hence, the same image prior can be used for both image deblurring and image denoising (*cf.* Chapter 4). Also note that denoising is a special case of non-blind deblurring if we assume additive white Gaussian noise in both cases (*cf.* Section 1.2).

DENOISING Compared to image deblurring (see below), image denoising is typically an easier problem since there is no additional corruption besides the noise. Furthermore, noise is typically assumed to be independent at each pixel of the image. While other noise assumptions can be made (*e. g.*, Poisson noise in low-light photography [Chatterjee et al., 2011] or medical imaging [Rodrigues et al., 2008]), we focus on additive Gaussian noise in this dissertation, which is the dominant noise assumption in the literature.

Optimization-based approaches have been very successful, such as the influential *Rudin-Osher-Fatemi* [Rudin et al., 1992] approach based on regularization via *total variation*. Also well-known are *nonlinear diffusion* methods [*cf.* Weickert, 1997], starting with the *Perona-Malik* model [Perona and Malik, 1990]. Another approach is to use *wavelet* *denoising* [*e.g.*, Portilla et al., 2003] by changing (*e.g.*, thresholding) the wavelets coefficients that represent higher frequencies of an image. An entirely different strategy is to exploit *self-similarity* within an image [Buades et al., 2005; Dabov et al., 2007b]. We will discuss some of these approaches in more detail in Section 2.6.3.3.

In the context of graphical models, Roth and Black [2009] propose the influential high-order FoE prior and apply it to image denoising by performing MAP estimation; further improved results are obtained by making use of discriminative training [Samuel and Tappen, 2009; Chen et al., 2013]. Schmidt et al. [2010] and Gao and Roth [2012] improve generative training to obtain FoE image priors that yield improved denoising results via minimum mean squared error (MMSE) estimation.

DEBLURRING Image blur (*e.g.*, camera shake) is one of the main sources of image corruption in digital photography and hard to undo. Image deblurring has thus been an active area of research, going back to the pioneering works of Wiener [1949], Richardson [1972], and Lucy [1974]. Recent work has predominantly focused on *blind deblurring* [*e.g.*, Fergus et al., 2006; Yuan et al., 2007; Joshi et al., 2008; Shan et al., 2008; Cho and Lee, 2009; Xu and Jia, 2010; Levin et al., 2011], particularly on estimating the blur from images (stationary and non-stationary [Whyte et al., 2010]). However, the problem of *non-blind deblurring* is an important component of many blind deblurring methods. While some approaches jointly predict the blur and the restored image [*e.g.*, Shan et al., 2008], it is sensible and common to separate the deblurring problem into first estimating the blur from the observed image, and then performing non-blind deblurring to obtain the restored image [*cf.* Levin et al., 2009]. Furthermore, non-blind deblurring is also applied when the blur is known [*e.g.*, Levin et al., 2007] or estimated by other means, such as special hardware [Ben-Ezra and Nayar, 2004; Joshi et al., 2010; Tai et al., 2008].

The Lucy-Richardson method [Lucy, 1974; Richardson, 1972] for non-blind deblurring is a classic and well-known approach. Although its performance is sub-par for natural images [*cf.* Levin et al., 2009], it is frequently used as a baseline [Krishnan and Fergus, 2009; Shan et al., 2008] and also for images with different properties (*e.g.*, in microscopy [Temerinac-Ott et al., 2012] or for low-light images [Whyte et al., 2014; Hu et al., 2014]). Also very common is to use manually-defined image priors formulated as MRFs with sparse, *i.e.* non-Gaussian, potential functions [Levin et al., 2007; Krishnan and Fergus, 2009; Xu and Jia, 2010]. Learning-based approaches had been restricted to generatively trained models [Schmidt et al., 2011]; discriminative deblurring methods have only recently been introduced (Chapter 6).

2.6.2 *Evaluation*

While image restoration methods can differ in various ways, the quality of the restored image is often of utmost importance. Hence, evaluation focuses on comparing restored images in *qualitative* (*i. e.*, subjective) and *quantitative* (*i. e.*, objective) ways. Qualitative comparisons typically highlight certain regions in the image that one method was able to (subjectively) better restore than others. Quantitative comparisons are based on *image quality measures* (explained below), which assign an objective score to the restored image; however, this typically requires access to the true uncorrupted image, called *ground truth* (GT), which is unavailable in practice. Hence, quantitative evaluation is mostly based on artificially corrupted images, where the observed image is synthetically created from a clean image according to the assumed corruption process. In contrast, qualitative comparisons can also be carried out on real images encountered in practical applications. Also, quantitative comparisons are typically averaged over a set of *test images*, which have not been used during model training. Reporting of average results is done to ensure that methods generalize well to a wider range of images, not only few (possibly cherry-picked) images.

2.6.2.1 *Datasets*

There are 5 standard test images called *Lena*, *Barbara*, *Boats*, *House*, and *Peppers*, which are still often used (mostly for qualitative comparisons) in the image processing and computer vision literature [*cf.* Portilla et al., 2003, Appendix B].

Clean images from the *Berkeley segmentation dataset* (BSDS) are frequently used to create artificially corrupted versions that are employed for model training and quantitative evaluation. This started with Roth and Black [2009], who used a subset of 68 images to quantitatively compare the denoising results from their FoE model to other image denoising methods. This test dataset has since then been adopted by many authors [*e. g.*, Samuel and Tappen, 2009; Barbu, 2009; Schmidt et al., 2010; Zoran and Weiss, 2011; Chen et al., 2013] and has become a benchmark for comparing various image models.

In contrast to image denoising, where images with real noise are rarely used, image deblurring methods are frequently evaluated with real blurred images. While the comparisons are mostly qualitative [*e. g.*, Fergus et al., 2006], quantitative evaluation has recently become more popular with the advent of new benchmarks [Levin et al., 2009; Köhler et al., 2012].

SYNTHETIC DATA Although there are some standard artificially created datasets, such as the 68 images from [Roth and Black, 2009], care must be taken to ensure fair comparisons. Since each synthetic image is created *randomly* according to the assumed corruption process, the use of different random numbers will result in a different image. Unfortunately, the synthetic images (or random numbers used for synthesis) are sometimes not available. Furthermore, although the clean GT images are typically quantized, *i.e.* each pixel can only take one of several discrete values, the created synthetic images are often not quantized (*cf.* experiments in Chapters 6 and 7).

2.6.2.2 *Image quality measures*

We now discuss image quality measures that are commonly used for quantitative comparisons in the context of image restoration. We already mentioned in passing (Section 2.3.1) that the 0-1 loss is not suitable, since all images that do not exactly correspond to the correct solution are considered equally bad, *i.e.* are assigned the same loss. It makes more sense to use a "softer" loss functions that assigns a cost based on a *distance* to the correct solution. Coming up with a simple, but sensible distance between the correct image $\mathbf{x}$ and a prediction $\tilde{\mathbf{x}}$ is not difficult here, since we assume $\mathbf{x}, \tilde{\mathbf{x}} \in \mathbb{R}^D$ to be part of Euclidean D-space. Hence, based on the Euclidean distance $d(\tilde{\mathbf{x}}, \mathbf{x}) = \|\tilde{\mathbf{x}} - \mathbf{x}\|$, the *mean squared error* (MSE)

$$\mathrm{MSE}(\tilde{\mathbf{x}}, \mathbf{x}) = \frac{1}{D}\|\tilde{\mathbf{x}} - \mathbf{x}\|^2 = \frac{1}{D}\sum_{i=1}^{D}(\tilde{x}_i - x_i)^2 \tag{2.45}$$

is often used to denote the distance between two images; note that using the average squared error makes it easy to compare errors for images of different sizes.

Additionally, dividing the images by scalar R will allow us to normalize the value range of each pixel to lie between 0 and 1; this accounts for different maximum values that images may have (*e.g.*, $R = 255$ for typical 8-bit quantized images). Furthermore, to measure image quality – where larger values denote higher restoration quality – we invert the distance (error) between two images. Lastly, we use a logarithmic scale to make the quality measure useful from very small up to very large errors. Overall, this yields the widely-used *peak signal-to-noise ratio* (PSNR)

$\log_{10} u = \frac{\log u}{\log 10}$

$$\begin{aligned}\mathrm{PSNR}(\tilde{\mathbf{x}}, \mathbf{x}) &= 10 \cdot \log_{10}\left(\mathrm{MSE}(\tilde{\mathbf{x}}/R, \mathbf{x}/R)^{-1}\right) \\ &= 10 \cdot \log_{10}\left(\frac{R^2}{\mathrm{MSE}(\tilde{\mathbf{x}}, \mathbf{x})}\right) \\ &= 20 \cdot \log_{10}\left(\frac{R}{\sqrt{\mathrm{MSE}(\tilde{\mathbf{x}}, \mathbf{x})}}\right),\end{aligned} \tag{2.46}$$

which is measured in *decibels* (dB). We can expect $\mathrm{MSE}(\tilde{\mathbf{x}}, \mathbf{x}) \in [0, R^2]$ under intended usage, hence PSNR values range from 0 for maximally dissimilar images up to infinity for identical images. Although quite simple in its definition, PSNR often provides a reasonable approximation to human perception of image restoration quality.

Sometimes [*e. g.*, Portilla et al., 2003; Roth and Black, 2009], PSNR is alternatively defined as

$$\text{PSNR}_{\text{SD}}(\tilde{\mathbf{x}}, \mathbf{x}) = 20 \cdot \log_{10}\left(\frac{R}{\text{SD}(\tilde{\mathbf{x}} - \mathbf{x})}\right) \tag{2.47}$$

based on the (sample) standard deviation

For better comparison, we do not use Bessel's correction *for estimating the standard deviation.*

$$\text{SD}(\tilde{\mathbf{x}} - \mathbf{x}) = \sqrt{\frac{1}{D}\sum_{i=1}^{D}((\tilde{x}_i - x_i) - \mu)^2} = \sqrt{\text{MSE}(\tilde{\mathbf{x}} - \mu, \mathbf{x})} \tag{2.48}$$

with (sample) mean

$$\mu = \frac{1}{D}\sum_{i=1}^{D}\tilde{x}_i - x_i. \tag{2.49}$$

Note that the two different PSNR definitions (*i. e.*, Eqs. 2.46 and 2.47) are identical if $\mu = 0$. Otherwise, they are related as:

$$\text{PSNR}_{\text{SD}}(\tilde{\mathbf{x}}, \mathbf{x}) = \max_{b} \text{PSNR}(\tilde{\mathbf{x}} - b, \mathbf{x}) = \text{PSNR}(\tilde{\mathbf{x}} - \mu, \mathbf{x}). \tag{2.50}$$

Hence, the PSNR definition in Eq. (2.47) is invariant to a constant *additive* prediction bias, such as general tendency to over- or underestimate the values of all pixels. Invariances to other properties are also sometimes used, such as $\arg\max_c \text{PSNR}(c \cdot \tilde{\mathbf{x}}, \mathbf{x})$ for a constant *multiplicative* prediction bias [Köhler et al., 2012]. Since PSNR is pervasive in the image restoration literature, it is often not even explicitly defined; hence, one has to be careful to guarantee fully accurate comparisons.

We are typically interested in improving the human perception of restored images. In practice, we use approximations in the form of image quality measures, such as PSNR, since human perception is difficult to measure. Compared to PSNR, more sophisticated quality measures have been proposed in the literature [*e. g.*, Wang et al., 2004; Sheikh and Bovik, 2006; Chandler and Hemami, 2007]. Of these, fairly widely used in the context of image restoration is the *structural similarity* (SSIM) index [Wang et al., 2004], which assigns the restored image a score from 0 up to 1. Still, image quality measures are very simple compared to the human visual system. For example, the method by [Cho et al., 2012] preserves strong texture in appropriate regions (*e. g.*, bushes or fur), which led to lower PSNR and SSIM scores, but was generally preferred by users.

Note that PSNR and SSIM are instances of so-called *full-reference* image quality measures, *i. e.* they assume that the clean GT image is available. However, this is typically not the case in practice, hence there are also *no-reference* (or *blind*) quality measures that do not need access to the clean reference image [*e. g.*, Wang et al., 2002; Brandão and Queluz, 2008].

2.6.3 *Other related work*

We already discussed in some detail how MRFs and CRFs can be applied to image restoration problems. In this section, we briefly survey some related work on image restoration and highlight similarities and differences to our MRF-based modeling approach.

2.6.3.1 *Global methods*

With an MRF we obtain a *global* image model (for images of arbitrary size) by specifying only *local* interactions via clique potentials. Due to our assumption of translation-invariance, the potentials are the same regardless of their location in the image. The potentials (or experts) typically model filter responses (*cf.* Section 2.2.2), which can be obtained by computing the convolution of the image and the respective filters. As a result, such models are also called *convolutional*. A widely-used model class are *convolutional neural networks* (CNNs) [*cf.* LeCun et al., 2010], which essentially perform regression. Hence, they can be used as prediction function (instead of Eq. 2.40) in the context of deterministic inference and learning (*cf.* Section 2.4). Although CNNs are arguably most often used for classification [*e.g.*, Krizhevsky et al., 2012], they have also been applied to image restoration [*e.g.*, Jain and Seung, 2009].

We presented MRFs as models for *spatially-discrete* images with a fixed number of pixels. On the other hand, there is the distinct class of *variational approaches* [Rudin et al., 1992; Schnörr et al., 1996] that model images as *spatially-continuous* functions. However, variational approaches require discretization in order to be used in practice on a computer with finite resources. It has been shown [Szeliski, 1990; Schelten and Roth, 2011] that in some cases these seemingly disparate model classes can be quite similar or even equivalent.

2.6.3.2 *Local methods*

Let us assume an MRF that models maximal cliques of size $m \times m$ pixels, such as the FoE model in Section 2.2.2. If we restrict the MRF to images of $m \times m$ pixels only, then we are not making any conditional independence assumptions (the associated GM is fully-connected). Hence, calling the model "MRF" is actually inappropriate, since we are not making any Markov assumption. Instead, such models are typically called (image) *patch* models, since they can only reasonably be used to model image patches of small sizes (typically $m < 20$).

Products of experts [Hinton, 2002], that we use as clique potentials in the FoE (Eq. 2.14), have originally been proposed as patch models. Another possibility to model image patches of a fixed size is to use a Gaussian mixture model (GMM), which has been pursued by Zoran and Weiss [2011]. In a discriminative context, deep neural networks in the form of *multilayer perceptrons* (MLPs) have been used as regressors to restore corrupted image patches [Burger et al., 2012; Schuler et al., 2013].

In practice, we have to apply a *local* patch model of fixed size to large images of arbitrary sizes. To that end, simpler problems like image denoising can directly be handled by using the patch model to restore all overlapping regions of the large observed image; afterwards, for a given pixel of the large image, the results from all overlapping restored regions are simply averaged. This strategy has been used by Burger et al. [2012] for image denoising with an MLP-based patch regressor. Schuler et al. [2013] extended this approach to image deconvolution by first recasting the deconvolution problem to a denoising problem. After training a generative patch model, another possibility is to use it like a clique potential in an MRF, which has been proposed under the name *expected patch log likelihood* (EPLL) by Zoran and Weiss [2011], who applied it to image denoising, deblurring, and inpainting.

2.6.3.3 *Self-similarity*

Another interesting property of natural images is *self-similarity* [*cf.* Zontak and Irani, 2011], *i. e.* structures in a particular image are likely to repeat itself in similar form (*e. g.*, at different locations, scales, or orientations). Examples include continuous edges at the contours of objects, textured regions with repetitive patterns, or similar objects at different scales (*e. g.*, due to different distances to the camera).

Self-similarity can be useful for many applications. For example, Barnes et al. [2009] proposed the *PatchMatch* algorithm for interactive image editing, which is based on quickly finding approximate nearest-neighbor matches for a given image patch. PatchMatch has been further generalized to find arbitrary correspondences [Barnes et al., 2010], which has been exploited for applications like stereo matching [*e. g.*, Bleyer et al., 2011] or optical flow [*e. g.*, Hornáček et al., 2014].

In the context of image restoration, Buades et al. [2005] proposed the *non-local means* image denoising method, which effectively restores each pixel of the image by weighted averaging of other similar pixels; given the image patch around a reference pixel, the weights for all other pixels are determined by the distances of their surrounding image patches to the reference patch. Arguably, the most well-known image denoising approach based on self-similarity is *block-matching and 3D filtering* (BM3D) [Dabov et al., 2007b], which first gathers similar image regions into 3D groups, which are then processed (via *collaborative filtering*) to remove noise while preserving salient image structures. Applying BM3D and other methods based on self-similarity to image deblurring is more difficult, though. A typical strategy [*cf.*

Schuler et al., 2013] is to first apply a regularized inversion of the blur matrix, which however causes unwanted artifacts since image noise is amplified and correlated; removing these artifacts in a second step is then addressed as a (structured) denoising problem [*e.g.*, Danielyan et al., 2012].

Properties of a particular image (such as self-similarity) are sometimes referred to as *internal* statistics, whereas properties of a large dataset of images are called *external* statistics. Our MRF-based image priors thus model external statistics that apply to images in general, whereas non-local means and BM3D make use of internal statistics that only apply to a given single image. Sun and Tappen [2011] have made an attempt at combining internal an external statistics with a *non-local range MRF*, where each potential is not only based on pixels belonging to a local image patch (clique), but also pixels from other similar patches. Another approach is to use internal and external methods separately and then (learn to) combine the resulting outputs [*e.g.*, Jancsary et al., 2012a; Mosseri et al., 2013; Burger et al., 2013] to obtain an improved restored image that ideally retains the benefits of both approaches.

2.6.3.4 *Dictionary methods with sparse coding*

In Section 2.2, we defined the clique potentials of MRF image priors based on the assumption of *sparse* filter responses, *i.e.* response values are very small most of the time. The general concept of sparsity can be attributed to many approaches in the literature, where an image (or general signal) is first transformed into a different representation where most values are (close to) zero; the second step consists of choosing an appropriate model for the sparse representation.

In the context of natural images, *wavelet* decompositions [*cf.* Mallat, 2009] are well-known to yield sparse representations. For example, Portilla et al. [2003] model multi-scale wavelet representations of images locally with Gaussian scale mixtures (GSMs) and demonstrate the merits of the resulting *BLS-GSM* model in the context of image denoising. In general, decomposing an image (patch) as a linear combination of (only a few) elements from a *dictionary* is called *sparse coding*. One may think of this as representing the image vector in a new (overcomplete) basis, such that there are only few non-zero coefficients w.r.t. the new basis elements.

While sparse coding has been used with a-priori fixed dictionaries (*e.g.*, using wavelets), more recent works, *e.g.* the *K-SVD* approach [Elad and Aharon, 2006], have shown that it is beneficial to learn the dictionary from the images at hand in a given application. The collaborative filtering step to denoise groups of similar patches in BM3D is based on sparse coding with a fixed dictionary; the *LSSC* method [Mairal et al., 2009] extends this by learning the dictionary and enforcing that similar patches also admit similar sparse representations.

3

HALF-QUADRATIC INFERENCE

CONTENTS

3.1 Gaussian MRFs 46
3.2 Half-quadratic Augmentation 47
3.3 Envelope and Integral Type 49
  3.3.1 Envelope type 49
  3.3.2 Integral type 51
  3.3.3 Summary 55
3.4 Additive and Multiplicative Form 56
  3.4.1 Multiplicative form 58
  3.4.2 Additive form 67
  3.4.3 Summary 77
3.5 Solving Equation Systems 78
  3.5.1 Matrix factorizations 80
  3.5.2 Iterative solvers and preconditioners 83
  3.5.3 Sampling 86

This chapter provides a unifying review of *half-quadratic* (HQ) inference with MRF-based image models, specifically the Field of Experts (FoE) model that subsumes many MRFs from the literature (Section 2.2.2). We focus on probabilistic posterior prediction in a generative context, concretely MAP estimation and approximating posterior expectations via samples. However, our exposition also applies to CRFs and further includes drawing samples from the MRF prior as a special case (which is needed for approximate maximum likelihood learning, Section 2.3.2). Note that our discussion of MAP estimation carries over to energy minimization (Eq. 2.40) in a deterministic setting (Section 2.4).

For the remainder of this chapter, we assume the generic Gaussian likelihood

$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2\mathbf{I}), \tag{3.1}$$

which can model several image restoration problems including denoising (Eq. 1.2) and deblurring (Eq. 1.4). Furthermore, we use the FoE image prior

$$p(\mathbf{x}) \propto \prod_{c \in \mathcal{C}} \prod_{i=1}^{N} \exp\left(-\rho_i(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)})\right) \tag{3.2}$$

with N linear filters $\mathbf{f}_i$ and their associated penalty functions ρ_i.

As mentioned in Section 1.5, inference is comparatively simple under the assumption of a Gaussian image prior. To explain this in more detail, let us assume quadratic penalty functions $\rho_i(u) = \frac{\alpha_i}{2} u^2$, which lead to a Gaussian MRF image prior

Note that we omit the limits and write $\prod_i$ instead of $\prod_{i=1}^N$ ($\sum_i$ instead of $\sum_{i=1}^N$) for the remainder of this chapter.

$$p(\mathbf{x}) \propto \prod_{c \in \mathcal{C}} \prod_{i=1}^{N} \mathcal{N}\left(\mathbf{f}_i^\mathsf{T} \mathbf{x}_{(c)}; 0, \alpha_i^{-1}\right) = \mathcal{N}(\mathbf{x}; \mathbf{0}, \mathbf{\Sigma}_\mathbf{x}), \tag{3.3}$$

where $\mathbf{\Sigma}_\mathbf{x} = \mathbf{\Omega}_\mathbf{x}^{-1}$ with easily accessible *precision* matrix $\mathbf{\Omega}_\mathbf{x}$. Before going into detailed derivations later, we first give an intuition why Gaussian models are desirable for computational reasons. For now, it is only important to note that $\mathbf{\Omega}_\mathbf{x}$ is a sparse matrix whose non-zero entries are determined by the connections in the underlying graphical model (GM). The posterior distribution (via Bayes' theorem)

$$\begin{aligned} p(\mathbf{x}|\mathbf{y}) &\propto \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2 \mathbf{I}) \cdot \mathcal{N}(\mathbf{x}; \mathbf{0}, \mathbf{\Sigma}_\mathbf{x}) \\ &\propto \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}}, \mathbf{\Sigma}_{\mathbf{x}|\mathbf{y}}) \\ &\propto \mathcal{N}(\mathbf{x}; \mathbf{\Omega}_{\mathbf{x}|\mathbf{y}}^{-1} \boldsymbol{\eta}_{\mathbf{x}|\mathbf{y}}, \mathbf{\Omega}_{\mathbf{x}|\mathbf{y}}^{-1}) \end{aligned} \tag{3.4}$$

can be written as a multivariate Gaussian distribution with vector $\boldsymbol{\eta}_{\mathbf{x}|\mathbf{y}}$ and sparse precision matrix $\mathbf{\Omega}_{\mathbf{x}|\mathbf{y}}$, which both are easy to compute.

Computing the MAP estimate $\hat{\mathbf{x}} = \arg\max_\mathbf{x} p(\mathbf{x}|\mathbf{y})$ then consists of finding the mode of the Gaussian distribution as

$$\arg\max_\mathbf{x} p(\mathbf{x}|\mathbf{y}) = \arg\max_\mathbf{x} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}}, \mathbf{\Sigma}_{\mathbf{x}|\mathbf{y}}) \tag{3.5}$$

$$= \arg\min_\mathbf{x} \mathbf{x}^\mathsf{T} \mathbf{\Sigma}_{\mathbf{x}|\mathbf{y}}^{-1} \mathbf{x} - 2\mathbf{x}^\mathsf{T} \mathbf{\Sigma}_{\mathbf{x}|\mathbf{y}}^{-1} \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}} \tag{3.6}$$

$$= \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}}, \tag{3.7}$$

which corresponds to a quadratic optimization problem whose solution is well-known to be the Gaussian mean. In practice, the MAP estimate $\hat{\mathbf{x}} = \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}} = \mathbf{\Omega}_{\mathbf{x}|\mathbf{y}}^{-1} \boldsymbol{\eta}_{\mathbf{x}|\mathbf{y}}$ can be obtained by solving a (sparse) system of linear equations (*cf.* Section 3.5), which is computationally much cheaper than inverting the precision matrix. Note that drawing samples from $p(\mathbf{x}|\mathbf{y})$ can be done similarly, which will be explained in Section 3.5.3.

Gaussian distributions have appealing properties, but are unfortunately unsuitable as image priors. Recall from Section 2.2.1 that the statistics of natural images are distinctly heavy-tailed (and thus non-Gaussian). As a result, using image priors with Gaussian potentials (*i. e.*, quadratic penalties for filter responses as in Eq. 3.3) will favor images without sharp edges (large intensity jumps). Hence, we cannot restore images that exhibit sharp transitions at object discontinuities. However, we can use Gaussians as building blocks of more expressive models, such as commonly-used *Gaussian mixture models* (GMMs)

$$p(\mathbf{x}) = \sum_i \pi_i \mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_i, \mathbf{\Sigma}_i) \tag{3.8}$$

with mixture weights π_i. It is useful to interpret Gaussian mixtures as *latent variable models* with $p(\mathbf{x}) = \sum_z p(\mathbf{x}, z)$, where the conditional distribution $p(\mathbf{x}|z)$ is Gaussian given a fixed value for latent variable z (indicating or selecting a particular mixture component). Recasting non-Gaussian MRFs as *latent Gaussian models* can be seen as a core concept behind half-quadratic inference.

3.2 HALF-QUADRATIC AUGMENTATION

We will assume from now on (heavy-tailed) non-Gaussian potentials $\exp(-\rho_i(u))$ with *even* penalty functions ρ_i that are continuous everywhere and at least once differentiable (except at 0); however, we do not require them to be *convex* functions. Hence, this includes commonly-used potentials, such as Student-t or hyper-Laplacian (*cf.* Section 2.2.1 and Fig. 2.3(b)). In an attempt to retain the benefits of Gaussian inference, we can *locally* approximate each penalty ρ_i for every filter response u with a quadratic function. Specifically, we can choose $\phi_i(u, z)$ to be a quadratic approximation of $\rho_i(u)$, with fixed z determined by the current value of u. By carrying out such an approximation for all penalties, we obtain a *globally* quadratic energy and thus Gaussian posterior distribution.

Strictly speaking, all $\exp(-\rho_i(u))$ *are* expert *functions* (cf. *Section* 2.2.2*).*

A function $\rho : \mathbb{R} \to \mathbb{R}$ *is* even *if* $\forall x \in \mathbb{R} : \rho(x) = \rho(-x)$.

Such an approximation is at the core of *half-quadratic* regularization [Geman and Yang, 1995; Geman and Reynolds, 1992; Charbonnier et al., 1994], which aims to ease inference (*e.g.*, MAP estimation) by introducing (independent) auxiliary/latent variables z_{ic} for each filter and image clique, such that the prior is retained by performing an operation $\bigoplus \in \{\max, \sup, \sum, \int\}$ that eliminates the auxiliary variables:

$$p(\mathbf{x}) \propto \prod_{c \in \mathcal{C}} \prod_i \bigoplus_{z_{ic}} \exp\left(-\phi_i(\mathbf{f}_i^\mathsf{T} \mathbf{x}_{(c)}, z_{ic})\right). \tag{3.9}$$

Since multiplication is distributive over the operation $\bigoplus$, we can define an augmented prior as

$$p(\mathbf{x}, \mathbf{z}) \propto \prod_{c \in \mathcal{C}} \prod_i \exp\left(-\phi_i(\mathbf{f}_i^\mathsf{T} \mathbf{x}_{(c)}, z_{ic})\right) \tag{3.10}$$

with

$$p(\mathbf{x}) \propto \bigoplus_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}). \tag{3.11}$$

The name "half-quadratic" stems from the fact that $\phi_i(u, z)$ is quadratic in u when z is held fixed. This further implies that for a fixed setting of $\mathbf{z}$ the distribution $p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{\mathbf{x}|\mathbf{z}}, \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{z}})$ is jointly Gaussian. The augmented model $p(\mathbf{x}, \mathbf{z})$ can be thought of as a hierarchical graphical model (Fig. 3.1), where all factors of the MRF are Gaussian when the auxiliary variables $\mathbf{z}$ are known. When combined with a

Specific choices for $\phi_i(u, z)$ *are discussed in Section* 3.4*.*

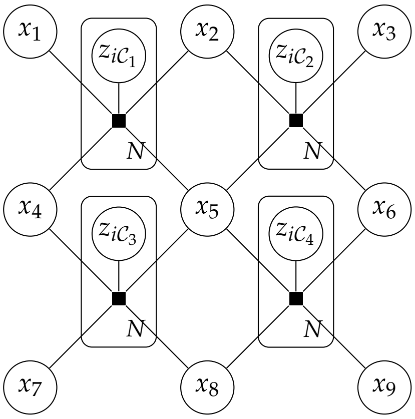


Figure 3.1: **Factor graph for augmented HQ image prior $p(\mathbf{x},\mathbf{z})$.** The GM is shown for an image of 3×3 pixels with filters $\mathbf{f}_i$ of size 2×2.

Gaussian likelihood, we obtain a Gaussian posterior for a fixed setting of $\mathbf{z}$:

$$\begin{aligned} p(\mathbf{x}|\mathbf{y},\mathbf{z}) &\propto \mathcal{N}(\mathbf{y};\mathbf{K}\mathbf{x},\sigma^2\mathbf{I}) \cdot \mathcal{N}(\mathbf{x};\boldsymbol{\mu}_{\mathbf{x}|\mathbf{z}},\boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{z}}) \\ &\propto \mathcal{N}(\mathbf{x};\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y},\mathbf{z}},\boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y},\mathbf{z}}). \end{aligned} \tag{3.12}$$

The benefit is that inference can now be carried out on the augmented posterior $p(\mathbf{x},\mathbf{z}|\mathbf{y})$ by alternating between updating $\mathbf{x}$ via $p(\mathbf{x}|\mathbf{y},\mathbf{z})$ and using $p(\mathbf{z}|\mathbf{x},\mathbf{y})$ to update the auxiliary variables based on the operation $\bigoplus$. While it may by counter-intuitive that we have made the problem easier by introducing additional variables, each of these two steps is relatively easy, as compared to directly using $p(\mathbf{x}|\mathbf{y})$. Specifically, the main advantage is that $p(\mathbf{x}|\mathbf{y},\mathbf{z})$ is jointly Gaussian, which means that updating $\mathbf{x}$ (*e.g.*, $\mathbf{x} \leftarrow \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y},\mathbf{z})$ in case of MAP estimation) amounts to solving sparse systems of linear equations. Updating $\mathbf{z}$ based on $p(\mathbf{z}|\mathbf{y},\mathbf{x})$ is also easy, since all z_{ic} are independent:

$$p(\mathbf{z}|\mathbf{x},\mathbf{y}) \propto \prod_{c\in\mathcal{C}} \prod_{i} p(z_{ic}|\mathbf{x},\mathbf{y}) \tag{3.13}$$

$$p(z_{ic}|\mathbf{x},\mathbf{y}) \propto \exp\big(-\phi_i(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)}, z_{ic})\big). \tag{3.14}$$

Furthermore, even if updating z_{ic} is rather complicated, it is still just a one-dimensional problem and can thus be pre-computed for all sensible values and then quickly retrieved via a lookup table [Krishnan and Fergus, 2009].

LIKELIHOOD ASSUMPTION Note that $p(\mathbf{x}|\mathbf{y},\mathbf{z})$ in Eq. (3.12) is only Gaussian under the assumption of a (commonly-used) Gaussian likelihood (Eq. 3.1). However, if this assumption was violated, one may still be able to obtain a Gaussian posterior $p(\mathbf{x}|\mathbf{y},\mathbf{z},\mathbf{u})$ by applying the same HQ technique to also obtain a HQ likelihood model $p(\mathbf{y},\mathbf{u}|\mathbf{x})$ with additional latent variables $\mathbf{u}$ [*cf.* Black and Rangarajan, 1996]. However, we do not address this here.

Algorithm 3.1 HQ envelope type – MAP estimation

```
$\hat{\mathbf{x}}^{(0)} \leftarrow \mathbf{y}$
for $t \leftarrow 1$ to $T$ do                                  ▷ Terminate after $T$ steps
    for $i \leftarrow 1$ to $N$, $c \in \mathcal{C}$ do         ▷ $\hat{\mathbf{z}}^{(t)} \leftarrow \arg\max_{\mathbf{z}} p(\mathbf{z}|\hat{\mathbf{x}}^{(t-1)}, \mathbf{y})$
        $\hat{z}_{ic}^{(t)} \leftarrow \arg\max_{z_{ic}} p(z_{ic}|\hat{\mathbf{x}}^{(t-1)}, \mathbf{y})$
    $\hat{\mathbf{x}}^{(t)} \leftarrow \arg\max_{\mathbf{x}} p(\mathbf{x}|\hat{\mathbf{z}}^{(t)}, \mathbf{y})$      ▷ System of linear equations
return $\{\hat{\mathbf{x}}^{(T)}, \hat{\mathbf{z}}^{(T)}\}$
```

INTERACTING LATENT VARIABLES The augmented image prior $p(\mathbf{x}, \mathbf{z})$ may by thought of as a hierarchical graphical model (Fig. 3.1). Hence, one could go a step further and also connect the latent variables to obtain an even richer model, which would be reminiscent of a *line process* as proposed by Geman and Geman [1984] (*cf.* Section 3.4.1). Unfortunately, when the latent variables are connected in the graphical model, inference becomes more difficult and it is also not possible to retain a typical MRF prior through elimination of the latent variables. Although not being popular (presumably) for the above reasons, HQ models with interacting latent variables have been explored in the literature [*e.g.*, Black and Rangarajan, 1996; Idier, 2001]. While we think that it would be interesting to revisit such models, we do not consider them here.

3.3 ENVELOPE AND INTEGRAL TYPE

We now go into details on why it is even valid to use $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$ to do inference on $p(\mathbf{x}|\mathbf{y})$ and explore two HQ *types* based on different possibilities for the operation $\bigoplus$. Section 3.4 will discuss the two dominant modeling choices for auxiliary function ϕ, which lead to two HQ *forms*. Note that we distinguish between *form* and *type* on purpose, since the combination of two HQ types and two HQ forms will yield four HQ variants (*cf.* Table 3.1).

3.3.1 *Envelope type*

Most commonly, the penalty function $\rho(u) = \min_z \phi(u, z)$ is expressed as the minimum (or infimum) over function $\phi(u, z)$ w.r.t. z, which results in the operation $\bigoplus = \max$ being the maximum (or supremum) function:

$$\begin{aligned} p(\mathbf{x}) &\propto \max_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) \\ &\propto \prod_{c \in \mathcal{C}} \prod_i \max_{z_{ic}} \exp\left(-\phi_i(\mathbf{f}_i^\mathsf{T} \mathbf{x}_{(c)}, z_{ic})\right). \end{aligned} \tag{3.15}$$

We call this half-quadratic variant the *envelope type* (as do Polson and Scott [2016]), because the non-quadratic penalty is expressed as the envelope of quadratic functions (see Fig. 3.2(a) for an example).

To the best of our knowledge, only MAP estimation to find $(\hat{\mathbf{x}}, \hat{\mathbf{z}}) = \arg\max_{\mathbf{x},\mathbf{z}} p(\mathbf{x}, \mathbf{z}|\mathbf{y})$ has been carried out with this particular type. MAP estimation [Charbonnier et al., 1994] alternates between computing the maximizer of $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$ and $p(\mathbf{z}|\mathbf{x}, \mathbf{y})$, as shown in Alg. 3.1. This is appealing, because both steps are typically easy to carry out, as mentioned above; we will go into more detail when we discuss the specific HQ forms in Section 3.4. It is easy to see that at the end of the iterative optimization procedure the found (local) maximum of $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$ is also a (local) maximum of the posterior $p(\mathbf{x}|\mathbf{y})$ that we are actually interested in:

$$\begin{aligned} p(\hat{\mathbf{x}}, \hat{\mathbf{z}}|\mathbf{y}) &= \max_{\mathbf{x},\mathbf{z}} p(\mathbf{x}, \mathbf{z}|\mathbf{y}) \\ &= \max_{\mathbf{x}} \max_{\mathbf{z}} p(\mathbf{y}|\mathbf{x}) p(\mathbf{x}, \mathbf{z}) \\ &= \max_{\mathbf{x}} p(\mathbf{y}|\mathbf{x}) \max_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) \\ &= \max_{\mathbf{x}} p(\mathbf{y}|\mathbf{x}) p(\mathbf{x}) \\ &= \max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y}). \end{aligned} \tag{3.16}$$

3.3.1.1 *Simulated annealing*

MAP estimation does *coordinate ascent* on the posterior $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$ and can get stuck in local optima. To alleviate this, we can instead apply the stochastic variant of MAP estimation called *simulated annealing* [Černý, 1985; Kirkpatrick et al., 1983; Geman and Geman, 1984], where the name and analogy come from the annealing process which is used to make metals and other solid materials. Instead of always greedily moving to a state with higher (augmented) posterior probability as in Alg. 3.1, simulated annealing will also allow moving to a less probable state, since it may help to find an even better solution in the end. Here, this can be achieved by deriving a so-called *tempered distribution* from the augmented posterior via exponentiation with a temperature parameter, and then doing stochastic updates via sampling using the tempered distribution (Alg. 3.2). Initially, a high temperature $H_1 \gg 1$ is chosen such that the tempered distribution is more "flat" compared to the posterior $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$, thus it is more likely to also move to less probable posterior states. The temperature is (slowly) decreased after each iteration, such that moves to less probable posterior states occur less frequently. Eventually, a low temperature $0 \leq H_t \ll 1$ leads to a tempered distribution that is more "peaked" than the posterior, where it is rare to move to less probable posterior states. Hence, the algorithm converges to a local optimum that hopefully has higher posterior probability as compared to the one achieved via MAP estimation as in Alg. 3.1. This comes at the expense of higher computational cost, since simulated annealing typically requires many more iterations to be effective (*i. e.*, slow tem-

Algorithm 3.2 HQ – MAP estimation (simulated annealing)

```
Require: “annealing schedule” $H_1, \dots, H_T$ with $H_t > H_{t+1}$
$\hat{\mathbf{x}}^{(0)} \leftarrow \mathbf{y}$
for $t \leftarrow 1$ to $T$ do                                  ▷ Budget of $T$ steps
    $q(\mathbf{x},\mathbf{z}|\mathbf{y}) \leftarrow p(\mathbf{x},\mathbf{z}|\mathbf{y})^{(1/H_t)}$          ▷ Tempered distribution
    for $i \leftarrow 1$ to $N$, $c \in \mathcal{C}$ do            ▷ $\hat{\mathbf{z}}^{(t)} \sim q(\mathbf{z}|\hat{\mathbf{x}}^{(t-1)},\mathbf{y})$
        $\hat{z}_{ic}^{(t)} \sim q(z_{ic}|\hat{\mathbf{x}}^{(t-1)},\mathbf{y})$
    $\hat{\mathbf{x}}^{(t)} \sim q(\mathbf{x}|\hat{\mathbf{z}}^{(t)},\mathbf{y})$                 ▷ Sampling from Gaussian MRF
return $\left\{\hat{\mathbf{x}}^{(T)},\hat{\mathbf{z}}^{(T)}\right\}$
```

perature decrease) and the updates via sampling (*cf.* Sections 3.3.2.2 and 3.5.3) are also somewhat more complicated.

These might be reasons why simulated annealing is rarely used nowadays for the problems that we consider in this thesis. However, it was the inference method of choice for Geman *et al.* [Geman and Reynolds, 1992; Geman and Yang, 1995] when they introduced the HQ approach. Charbonnier et al. [1994] first proposed to use MAP estimation as in Alg. 3.1, which they named *ARTUR* and *LEGEND*, respectively, for the two HQ forms (*cf.* Section 3.4).

3.3.2 *Integral type*

In the literature, MAP estimation with the envelope type of the previous section is typically discussed in a non-probabilistic context as an energy minimization approach. In contrast, we now explain a half-quadratic variant that is predicated on a probabilistic interpretation.

We briefly mentioned Gaussian mixture models (GMMs) before as one popular way to build more complicated distributions by using Gaussians as building blocks. We follow this here and express

$$\exp(-\rho(u)) = \int \exp(-\phi(u,z))\, dz \quad \text{with} \tag{3.17}$$

$$\phi(u,z) = -\log\big(p(z) \cdot \mathcal{N}(u, \mu_z, \beta_z^{-1})\big) \tag{3.18}$$

as a GMM, where we integrate ($\bigoplus = \int$) over the latent variable z that indicates the mixture component. That is why we call this half-quadratic variant the *integral type* (as do Palmer et al. [2006]). Note that we consider arbitrary Gaussian mixtures for now; we will look at specific variants and their properties in Section 3.4. When z is chosen as a random variable with a continuous domain, we also consider infinite Gaussian mixture models. When z is a discrete random variable, we use summation ($\bigoplus = \sum$) instead of integration.

We can write the prior as a marginal distribution with this representation:

$$\begin{aligned} p(\mathbf{x}) &\propto \int p(\mathbf{x},\mathbf{z})\ d\mathbf{z} \\ &\propto \prod_{c\in\mathcal{C}}\prod_{i}\int \exp\left(-\phi_i(\mathbf{f}_i^{\mathsf{T}}\mathbf{x}_{(c)}, z_{ic})\right)\ dz_{ic}. \end{aligned} \tag{3.19}$$

As an aside, note that Eq. (3.19) is a product of GMMs and as such also one large GMM. Unfortunately, even when each z_{ic} was a discrete random variable with only two possible states, we cannot work directly with the resulting large GMM because it has exponentially many mixture components, which makes computing the mixture weights $p(\mathbf{z})$ intractable [*cf.* Ihler et al., 2004].

3.3.2.1 *Expectation maximization*

To maximize the posterior $p(\mathbf{x}|\mathbf{y}) = \int p(\mathbf{x},\mathbf{z}|\mathbf{y})\ d\mathbf{z}$, we can resort to the well-known *expectation maximization* (EM) algorithm [Dempster et al., 1977], where the latent variables $\mathbf{z}$ are treated as missing or unobserved data. We will give a brief summary of EM in this context, which closely follows [Minka, 1998] and [Dellaert, 2002].

Note that EM is a local optimization method and may only find a local optimum.

Expectation maximization is an iterative algorithm that can be used to estimate $\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$ by maximizing the log-posterior in our setting. Since directly maximizing $\log p(\mathbf{x}|\mathbf{y})$ is difficult, EM first constructs a lower bound $b(\mathbf{x}^{(t)}, q) \leq \log p(\mathbf{x}^{(t)}|\mathbf{y})$ around $\mathbf{x}^{(t)}$ (the current estimate of $\hat{\mathbf{x}}$), where $q(\mathbf{z})$ can be any proper probability distribution, *i.e.* $\int q(\mathbf{z})\,d\mathbf{z} = 1$ and $q(\mathbf{z}) \geq 0$ for all $\mathbf{z}$. The basic idea is to maximize such a lower bound of the log-posterior at each step of EM. In particular, the bound is derived by using *Jensen's inequality* [Jensen, 1906] as follows:

$$\log p(\mathbf{x}^{(t)}|\mathbf{y}) = \log \int q(\mathbf{z}) \frac{p(\mathbf{x}^{(t)},\mathbf{z}|\mathbf{y})}{q(\mathbf{z})}\ d\mathbf{z} \geq \int q(\mathbf{z}) \log \frac{p(\mathbf{x}^{(t)},\mathbf{z}|\mathbf{y})}{q(\mathbf{z})}\ d\mathbf{z} = b(\mathbf{x}^{(t)}, q) \tag{3.20}$$

Although the bound holds for any $q(\mathbf{z})$, we want to find the tightest possible bound by choosing $q(\mathbf{z})$ such that the bound is as large as possible at our current estimate $\mathbf{x}^{(t)}$. To find such a density q, one way is to rewrite the bound as follows:

$$b(\mathbf{x}^{(t)}, q) = \int q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x}^{(t)},\mathbf{y})p(\mathbf{x}^{(t)}|\mathbf{y})}{q(\mathbf{z})}\ d\mathbf{z} \tag{3.21}$$

$$= \int q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x}^{(t)},\mathbf{y})}{q(\mathbf{z})} + q(\mathbf{z}) \log p(\mathbf{x}^{(t)}|\mathbf{y})\ d\mathbf{z} \tag{3.22}$$

$$= -\int q(\mathbf{z}) \log \frac{q(\mathbf{z})}{p(\mathbf{z}|\mathbf{x}^{(t)},\mathbf{y})}\ d\mathbf{z} + \log p(\mathbf{x}^{(t)}|\mathbf{y}) \tag{3.23}$$

$$= -D_{\mathrm{KL}}\left(q(\mathbf{z}) \,\|\, p(\mathbf{z}|\mathbf{x}^{(t)},\mathbf{y})\right) + \log p(\mathbf{x}^{(t)}|\mathbf{y}) \tag{3.24}$$

Algorithm 3.3 HQ integral type – MAP estimation (EM)

```
1: $\hat{\mathbf{x}}^{(0)} \leftarrow \mathbf{y}$
2: for $t \leftarrow 1$ to $T$ do                                ▷ Terminate after $T$ steps
3:     for $i \leftarrow 1$ to $N$, $c \in \mathcal{C}$ do          ▷ $\hat{\mathbf{z}}^{(t)} \leftarrow \mathbb{E}[\mathbf{z}|\hat{\mathbf{x}}^{(t-1)}, \mathbf{y}]$
4:         $\hat{z}_{ic}^{(t)} \leftarrow \mathbb{E}[z_{ic}|\hat{\mathbf{x}}^{(t-1)}, \mathbf{y}]$
5:     $\hat{\mathbf{x}}^{(t)} \leftarrow \arg\max_{\mathbf{x}} p(\mathbf{x}|\hat{\mathbf{z}}^{(t)}, \mathbf{y})$       ▷ System of linear equations
6: return $\{\hat{\mathbf{x}}^{(T)}, \hat{\mathbf{z}}^{(T)}\}$
```

where $D_{\mathrm{KL}}(q\|p)$ denotes the non-negative *Kullback-Leibler divergence* [Kullback and Leibler, 1951] that is 0 if and only if the densities p and q are identical. Hence, the lower bound $b(\mathbf{x}^{(t)}, q) = \log p(\mathbf{x}^{(t)}|\mathbf{y})$ touches the log-posterior at $\mathbf{x}^{(t)}$ when we choose $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x}^{(t)}, \mathbf{y})$. Specifically, we will show that this corresponds to each potential of the image prior being tightly lower-bounded at $\mathbf{x}^{(t)}$ by a Gaussian distribution (*cf.* Figs. 3.4(a) and 3.7(a)). Computing $p(\mathbf{z}|\mathbf{x}^{(t)}, \mathbf{y})$ to obtain a good lower bound is called the "E-step". The subsequent "M-step" then consists of maximizing the bound w.r.t. $\mathbf{x}$:

$D_{KL}(a\|b) = \int a(x) \log \frac{a(x)}{b(x)}\, dx.$

$$\begin{aligned}
\mathbf{x}^{(t+1)} &= \arg\max_{\mathbf{x}} b\big(\mathbf{x}, p(\mathbf{z}|\mathbf{x}^{(t)}, \mathbf{y})\big) \\
&= \arg\max_{\mathbf{x}} \int p(\mathbf{z}|\mathbf{x}^{(t)}, \mathbf{y}) \log\big(p(\mathbf{y}|\mathbf{x})p(\mathbf{x}, \mathbf{z})\big)\, d\mathbf{z} \\
&= \arg\max_{\mathbf{x}} \log p(\mathbf{y}|\mathbf{x}) - \int p(\mathbf{z}|\mathbf{x}^{(t)}, \mathbf{y}) \sum_{c\in\mathcal{C}} \sum_{i} \phi_i(\mathbf{f}_i^{\mathrm{T}} \mathbf{x}_{(c)}, z_{ic})\, d\mathbf{z} \\
&= \arg\max_{\mathbf{x}} \log p(\mathbf{y}|\mathbf{x}) - \sum_{c\in\mathcal{C}} \sum_{i} \mathbb{E}_{z_{ic}} \left[\phi_i(\mathbf{f}_i^{\mathrm{T}} \mathbf{x}_{(c)}, z_{ic})|\mathbf{x}^{(t)}, \mathbf{y}\right] \qquad (3.25)
\end{aligned}$$

Since all terms are quadratic, we can find the value that maximizes the bound by solving a system of linear equations.

Furthermore, if we assume that the latent variables either determine *only* the variance or the mean of the Gaussian mixture (but not both), we can further simplify this. These two cases correspond to the half-quadratic forms that we will discuss in detail in Section 3.4. In the first case, we have $\phi(u,z) = \frac{z}{2}(u-\mu)^2 + \mathrm{const}(z)$, from which follows that

$$\mathbb{E}[\phi(u,z)] = \frac{\mathbb{E}[z]}{2}(u-\mu)^2 + \mathrm{const} = \phi(u, \mathbb{E}[z]) + \mathrm{const}. \qquad (3.26)$$

Here, $\mathrm{const}(z)$ denotes an arbitrary term that depends on z, but not u. In the second case, we have $\phi(u,z) = \frac{\beta}{2}(u-z)^2 + \mathrm{const}(z)$, from which follows that

$$\begin{aligned}
\mathbb{E}[\phi(u,z)] &= \frac{\beta}{2}\left(u^2 - 2u\mathbb{E}[z] + (\mathbb{E}[z])^2\right) + \mathrm{const} \\
&= \phi(u, \mathbb{E}[z]) + \mathrm{const},
\end{aligned} \qquad (3.27)$$

where we simply added the term $(\mathbb{E}[z])^2$ and absorbed others in the constant term. In both cases, we can now rewrite Eq. (3.25) as

$$\begin{aligned} \mathbf{x}^{(t+1)} &= \arg\max_{\mathbf{x}} \log p(\mathbf{y}|\mathbf{x}) - \sum_{c\in\mathcal{C}} \sum_i \phi_i(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)}, \mathbb{E}[z_{ic}|\mathbf{x}^{(t)}, \mathbf{y}]) \\ &= \arg\max_{\mathbf{x}} p(\mathbf{x}|\bar{\mathbf{z}}, \mathbf{y}) \end{aligned} \tag{3.28}$$

where $\bar{\mathbf{z}} = \mathbb{E}[\mathbf{z}|\mathbf{x}^{(t)}, \mathbf{y}]$. Hence, we can do EM-based MAP estimation by alternating between computing the expected value of the latent variables and maximizing the augmented posterior conditioned on the latent variables. The whole procedure is summarized in Alg. 3.3, which shows the similarity to MAP estimation in case of the envelope type (Alg. 3.1); the two algorithms only differ in the update of the latent variables (line 4). It can be shown that the two algorithms are indeed identical [Champagnat and Idier, 2004; Palmer et al., 2006], which we will illustrate in more detail in Section 3.4.

3.3.2.2 *Gibbs sampling*

Sampling from the prior $p(\mathbf{x})$ can be done similarly.

Using an EM algorithm to compute the MAP estimate $\arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$ is analogous to energy minimization used in deterministic modeling (Section 2.4). However, we often want to compute (more complicated) posterior expectations in the context of probabilistic inference (Section 2.3.1), which are made tractable through sampling-based approximations. To that end, we introduced the general concept of an auxiliary variable block Gibbs sampler (Section 2.3.1.1), which we now make more concrete for the augmented HQ posterior $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$ with auxiliary variables $\mathbf{z}$.

The Gibbs sampler will alternate between sampling from the conditional distributions $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$ and $p(\mathbf{z}|\mathbf{x}, \mathbf{y})$ (Alg. 3.4). This way, we can draw a set of samples from the joint distribution $\{\{\mathbf{x}^{(t)}, \mathbf{z}^{(t)}\}\}_{t=1}^T \sim p(\mathbf{x}, \mathbf{z}|\mathbf{y})$. Since we are typically not interested in the latent variables $\mathbf{z}$, we can simply discard all $\mathbf{z}^{(t)}$ samples; the remaining samples $\{\mathbf{x}^{(t)}\}_{t=1}^T \sim p(\mathbf{x}|\mathbf{y})$ are representative of the original posterior. Sampling from both of the conditional distributions is relatively easy. The latent variables are independent and can thus be sampled individually via univariate distributions:

When $p(z_{ic}|\mathbf{x}, \mathbf{y})$ does not have a closed-form expression, it can typically be discretized into a multinomial distribution.

$$p(z_{ic}|\mathbf{x}, \mathbf{y}) = \frac{\exp\left(-\phi_i(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)}, z_{ic})\right)}{\int \exp\left(-\phi_i(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)}, z_{ic})\right)\, dz_{ic}}. \tag{3.29}$$

Sampling from $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$ is somewhat more difficult than just finding its maximum w.r.t. $\mathbf{x}$, as we have done in the EM algorithm. However, since $p(\mathbf{x}|\mathbf{z}, \mathbf{y}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}}^{-1}\boldsymbol{\eta}_{\mathbf{x}|\mathbf{z},\mathbf{y}}, \boldsymbol{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}}^{-1})$ is a Gaussian distribution, we can still sample by solving equation systems that involve the sparse system matrix $\boldsymbol{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}}$ (and a factorization), which will be explained in detail in Section 3.5.

Algorithm 3.4 HQ – Gibbs sampling

```
Require: "burn-in" threshold $0 \leq B < T$
Initialize $\mathbf{x}^{(0)}$,
for $t \leftarrow 1$ to $T$ do                         ▷ Markov chain of length $T$
    for $i \leftarrow 1$ to $N$, $c \in \mathcal{C}$ do    ▷ $\mathbf{z}^{(t)} \sim p(\mathbf{z}|\mathbf{x}^{(t-1)}, \mathbf{y})$
        $z_{ic}^{(t)} \sim p(z_{ic}|\mathbf{x}^{(t-1)}, \mathbf{y})$
    $\mathbf{x}^{(t)} \sim p(\mathbf{x}|\mathbf{z}^{(t)}, \mathbf{y})$          ▷ Sampling from Gaussian MRF
return $\{\mathbf{x}^{(t)}, \mathbf{z}^{(t)}\}_{t=B}^{T}$
```

ENVELOPE TYPE Note that Gibbs sampling can also be applied in the envelope type, but the resulting samples drawn from $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$ cannot be used to approximate distributional properties of the posterior $p(\mathbf{x}|\mathbf{y})$ (that we are interested in), since

$$p(\mathbf{x}|\mathbf{y}) = \max_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}|\mathbf{y}) \neq \int p(\mathbf{x}, \mathbf{z}|\mathbf{y}) \; d\mathbf{z} \tag{3.30}$$

is not the marginal distribution of $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$. However, sampling from $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$ can be used for MAP estimation via simulated annealing (Alg. 3.2).

3.3.3 *Summary*

We have introduced the two main *types* of half-quadratic augmentations, namely the envelope and integral type. Both types lead to very similar MAP estimation algorithms (Algs. 3.1 and 3.3), which can actually shown to be identical [Champagnat and Idier, 2004; Palmer et al., 2006]. Intuitively, MAP estimation in both cases consists of first locally bounding each robust penalty with a quadratic function, then maximizing the bound. This is repeated until convergence or a desired accuracy. Hence, MAP estimation can be seen as a *local variational method* [*cf.* Bishop, 2006, § 10.5.]. We will show the equivalence for MAP estimation between the two types in more detail and with concrete examples after we introduce the two *forms* of half-quadratic representations in Section 3.4. Table 3.1 provides a brief overview of the HQ variants in this chapter.

If MAP estimation is our only goal, it is favorable to use the envelope type for several reasons, as we will show in the next section. However, if we are interested in probabilistic inference beyond the MAP estimate, it is essential to use the integral type, as it allows us to draw samples from the posterior distribution with an efficient (block) Gibbs sampler.

HQ type	HQ form	HQ representation of potential function
Envelope	Multiplicative	$\exp(-\rho(u)) = \max_z \exp(-\frac{1}{2}u^2 z - \psi(z))$
Envelope	Additive	$\exp(-\rho(u)) = \max_z \exp(-\frac{\beta}{2}(u-z)^2 - \psi(z))$
Integral	Multiplicative	$\exp(-\rho(u)) = \int \exp(-\frac{1}{2}u^2 z - \psi(z))\ dz$
Integral	Additive	$\exp(-\rho(u)) = \int \exp(-\frac{\beta}{2}(u-z)^2 - \psi(z))\ dz$

Table 3.1: Overview of the four HQ variants.

3.4 ADDITIVE AND MULTIPLICATIVE FORM

Thus far, we have not talked about specific penalty functions $\rho(u)$ and how (when possible) to obtain their half-quadratic counterparts $\phi(u,z)$, such that the relationship

$$\exp(-\rho(u)) = \bigoplus_z \exp(-\phi(u,z)) \tag{3.31}$$

holds. We have only assumed that $\phi(u,z)$ is a quadratic function in u when z is held fixed. However, whether z has an effect on the *scale* or *location* (or both) of the quadratic function does make a difference in the type of potentials that can be used, the rate of convergence of HQ inference, and the computational cost of each step of HQ inference.

To make this clearer, it may be easier to talk about this in probabilistic terms when considering the GMM

$$\exp(-\rho(u)) = \int \exp(-\phi(u,z))\ dz = \int p(z) \cdot \mathcal{N}(u, \mu_z, \beta_z^{-1})\ dz \tag{3.32}$$

of the integral type from Section 3.3.2. An arbitrary GMM can approximate any positive (multimodal) probability distribution arbitrarily well (in the limit of an infinite number of mixture components). However, the potential functions used in practice are not multimodal, they are even functions centered at 0. Hence, GMMs are typically used to model potential functions by using mixture components that all have the same mean, *i. e.* $\mu_z = \mu = 0$, where the latent variable z affects the scale (variance) of a mixture component, but not its location (mean); this is called a Gaussian scale mixture (GSM) (see example in Fig. 3.2(b)). Alternatively, we may define the GMM through mixture components that differ by their locations (selected through z), but all share the same scale, *i. e.* $\beta_z = \beta$; this is called a Gaussian location mixture (GLM) (see example in Fig. 3.5(b)). The latter variant is less common, but appealing under some circumstances as shown later.

GSMs and GLMs are representative of the two choices for the form of ϕ that have been discussed in the literature: the *multiplicative form* [Geman and Reynolds, 1992]

$$\phi(u,z) = \frac{1}{2}u^2 z + \psi(z) \tag{3.33}$$

and the *additive form* [Geman and Yang, 1995]

$$\phi(u,z) = \frac{\beta}{2}(u-z)^2 + \psi(z). \tag{3.34}$$

Geman and Yang [1995] only used $\beta = 1$ when they introduced the additive form. Idier [2001] considered a similar scaling approach to ours.

In either case, $\psi(z)$ must be chosen such that Eq. (3.31) is satisfied; an additional scaling parameter β is often used in the additive form. The names stem from the fact that the latent variable z either has a multiplicative or additive effect on the quadratic variable u, affecting scale or location of the quadratic approximation, respectively. In the literature, the additive form is sometimes abbreviated as *GY* (for Geman and Yang), and the multiplicative form as *GR* (for Geman and Reynolds).

Polson and Scott [2016] studied more general HQ forms that affect both scale and location, which are useful for asymmetric functions, e. g. *for likelihoods.*

Overall, we consider four HQ variants due to the combination of two HQ types and two HQ forms (Table 3.1). In the following, we will discuss these in more detail.

3.4.0.1 *Convex duality*

Before doing so, however, we will introduce the concept of *convex duality* [*cf.* Rockafellar, 1970, §12], which is the backbone of HQ forms of the envelope type. In particular, any concave function $f(v)$ can be uniquely represented as

$$f(v) = \min_z \{vz - f^*(z)\} \tag{3.35}$$

via its *conjugate* (or *dual*) function f^*, which itself can be expressed as

$$f^*(z) = \min_v \{vz - f(v)\}. \tag{3.36}$$

The function f^* is thus called the *concave conjugate* of f. (The *convex conjugate* is defined analogously where f is a convex function and minimization is replaced by maximization.)

The dual function is also known as Legendre-Fenchel *transformation.*

Intuitively, Eq. (3.35) describes the concave function f as being linearly upper bounded by all its tangents (with slope z). However, we will be interested in using the dual function to obtain not linear but quadratic bounds. To that end, we apply variable transformations, such as $v = u^2$, to obtain a quadratic upper bound on $f(v)$ via

$$f(u^2) = \min_z \left\{u^2 z - \bar{f}^*(z)\right\}, \tag{3.37}$$

where $\bar{f}^*(z)$ corresponds to the conjugate of $f(u^2)$ [*cf.* Jordan et al., 1999]. For this to apply, f must be a concave function of u^2. As we will show, by means of variable transformations, convex duality can be applied in both HQ forms to obtain quadratic bounds on penalty functions ρ.

Although the sign is lost through variable transformation $v = u^2$, this is not important since we assume all penalty functions to be even, i. e. *$\rho(u) = \rho(-u)$.*

Furthermore, the minimizer of Eq. (3.35) is well-known [*e. g.*, Boyd and Vandenberghe, 2004, § 3.3.2] to be the derivative $f'(v) = \frac{d}{dv}f(v)$, *i. e.*

$$f'(v) = \arg\min_z \{vz - f^*(z)\}, \tag{3.38}$$

which of course assumes that f is differentiable. This will be very useful to update the latent variables during MAP estimation (*cf.* Alg. 3.1).

3.4.1 *Multiplicative form*

Compared to the additive form, the multiplicative form is arguably more widespread and has received more attention in the literature. One possible reason is that it is more natural or intuitive to express a unimodal (potential) function as a combination of quadratic functions that are all centered around the mode of the target function. Furthermore, the multiplicative form is applicable to a wider variety of robust potentials used in practice [*cf.* Black and Rangarajan, 1996], as compared to the additive form.

3.4.1.1 *Envelope type*

We will first talk about the envelope type, where we choose $\bigoplus = \max$, hence

$$\rho(u) = \min_z \phi(u,z) = \min_z \left\{ \frac{1}{2}u^2 z + \psi(z) \right\}. \tag{3.39}$$

In order to find a function ψ to satisfy Eq. (3.39) and to know which conditions ρ has to satisfy for this to be applicable, we use convex duality properties as defined above. We first transform the equation

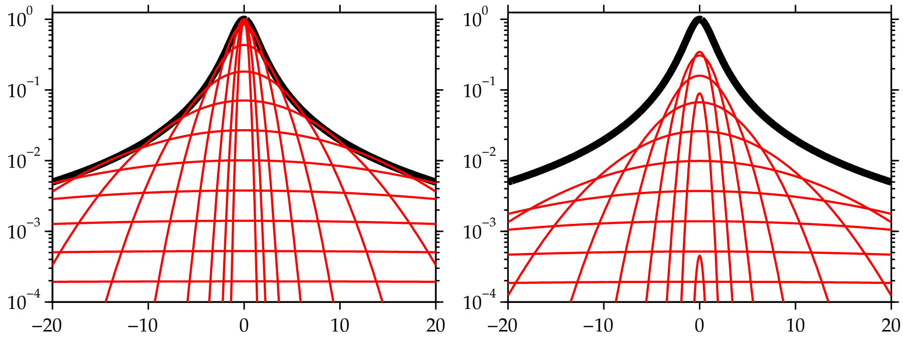
$$\begin{aligned} & & \rho(u) &= \min_z \left\{ \frac{1}{2}u^2 z + \psi(z) \right\} & (3.40)\\ \Leftrightarrow & & \rho(\zeta^{-1}(\zeta(u))) &= \min_z \left\{ \zeta(u)z + \psi(z) \right\} & (3.41)\\ \Leftrightarrow & & \rho(\sqrt{2v}) &= \min_z \left\{ vz + \psi(z) \right\} & (3.42) \end{aligned}$$

with the substitution $\zeta(u) = v = \frac{1}{2}u^2$ (and thus $\zeta^{-1}(v) = u = \sqrt{2v}$). If $\rho(\sqrt{2v})$ is concave for all $v \geq 0$, convex duality applies and we can obtain ψ to satisfy Eq. (3.39) via its concave conjugate as

$$\psi(z) = -\min_{v \geq 0} \left\{ vz - \rho(\sqrt{2v}) \right\} = -\min_u \left\{ \frac{1}{2}u^2 z - \rho(u) \right\}, \tag{3.43}$$

where we undid the variable substitution.

The above existence conditions of the (envelope type) multiplicative form of Eq. (3.33) have already been discussed by Geman and Reynolds [1992] when they introduced the HQ method. A generic recipe to convert between robust potential and half-quadratic representations has been given by Black and Rangarajan [1996]. Further details are discussed by many other authors [*e.g.*, Idier, 2001; Champagnat and Idier, 2004; Nikolova and Ng, 2005; Palmer et al., 2006; Nikolova and Chan, 2007; Polson and Scott, 2016].



(a) Envelope type (b) Integral type

Figure 3.2: **Multiplicative form (example).** HQ representation of a Student-t potential $\exp(-\rho(u))$ (thick black) with $\rho(u) = \log(1 + \frac{1}{2}u^2)$ and the associated $\exp(-\phi(u, z))$ for a few values of z (red). Integral type representation in *(b)* scaled for better comparison.

MAP ESTIMATION Furthermore, we can obtain the minimizer of Eq. (3.39) easily via computing the derivative of ρ, which has already been exploited by Charbonnier et al. [1994]. Let $f(v) = \rho(\sqrt{2v})$ and $f^*(z) = -\psi(z)$ be its dual, then we can use Eq. (3.38) to determine

$$\arg\min_z \left\{ \frac{1}{2}u^2 z + \psi(z) \right\} = f'(v) = \frac{d\rho(v)}{dv} \cdot \frac{1}{\sqrt{2v}} = \frac{\rho'(u)}{u}, \qquad (3.44)$$

where we have substituted back $v = \frac{1}{2}u^2$. The value of Eq. (3.44) at $u = 0$ can be implicitly computed by continuity of ρ as $\lim_{u \to 0} \rho'(u)/u$ [*cf.* Charbonnier et al., 1997; Idier, 2001; Champagnat and Idier, 2004].

Note that this result is important in practice, since Eq. (3.44) is actually the only equation we need for updating the latent variables z_{ic} during half-quadratic MAP estimation, *i. e.* line 4 of Alg. 3.1 can be carried out as follows (*cf.* Fig. 3.3):

$$\begin{aligned} \arg\max_{z_{ic}} p(z_{ic}|\mathbf{x}, \mathbf{y}) &= \arg\min_{z_{ic}} \phi_i(\mathbf{f}_i^{\mathsf{T}}\mathbf{x}_{(c)}, z_{ic}) \\ &= \arg\min_{z_{ic}} \left\{ \frac{1}{2}(\mathbf{f}_i^{\mathsf{T}}\mathbf{x}_{(c)})^2 z_{ic} + \psi_i(z_{ic}) \right\} \\ &= \frac{\rho_i'(\mathbf{f}_i^{\mathsf{T}}\mathbf{x}_{(c)})}{\mathbf{f}_i^{\mathsf{T}}\mathbf{x}_{(c)}}. \end{aligned} \qquad (3.45)$$

3.4.1.2 *Integral type*

In the integral type, we choose $\bigoplus = \int$, hence

$$\begin{aligned} \exp(-\rho(u)) &= \int \exp(-\phi(u, z)) \; dz \\ &= \int \exp(-\psi(z)) \cdot \exp\left(-\frac{z}{2}u^2\right) \; dz \\ &\propto \int p_\psi(z) \cdot \mathcal{N}(u; 0, z^{-1}) \; dz, \end{aligned} \qquad (3.46)$$

where $p_\psi(z)$ is the so-called *mixing distribution* of the latent variable z, which determines the precision (inverse variance) of the Gaussian *mixture distribution*. Eq. (3.46) corresponds to the well-known model class of Gaussian scale mixtures (GSMs) [Andrews and Mallows, 1974], to which several robust potentials, such as Student-t or (hyper-)Laplacian, belong [*cf.* Gneiting, 1997].

In the context of HQ models, the conditions when a distribution can be expressed as a GSM are for example discussed by Palmer et al. [2006] and Polson and Scott [2016]. Concretely, the potential $\varphi(u) = \exp(-\rho(u))$ can be represented as a GSM if and only if $\varphi(\sqrt{u})$ is *completely monotonic* on $(0, \infty)$ [Palmer et al., 2006, Thm. 3]. An infinitely-differentiable function $f(u)$ is completely monotonic on (a, b) if

$f^{(n)}(u) = \frac{d^n}{du^n} f(u)$

$$(-1)^n f^{(n)}(u) \geq 0, \quad n = 0, 1, \ldots \tag{3.47}$$

for every $u \in (a, b)$. This characterization of GSMs hinges on the work of Bernstein and Widder [Widder, 1946, Chapter IV, § 12]. Furthermore, it can be shown that when a potential can be represented in the integral type, then it can also be represented in the envelope type [Palmer et al., 2006; Polson and Scott, 2016]. In other words, the set of potentials representable in the integral type is a subset of potentials representable in the envelope type.

Unfortunately, besides the known existence conditions as stated above, to the best of our knowledge there is no simple recipe to convert a suitable potential to a GSM. This is contrast to the envelope type, where properties of convex duality can be used to easily derive the half-quadratic representation. However, we can go the opposite way and choose a mixing distribution and then obtain the associated GSM potential, which does not need to have a closed-form expression. Specifically, it is always possible to simply define a discrete (multinomial) mixing distribution over a fixed number of Gaussian mixture components. We will make use of this in Chapter 4.

MAP ESTIMATION However, if we only want to do MAP estimation, there is no need to actually find a GSM representation for the potential, since updating the latent variables during half-quadratic inference only depends on the penalty ρ. To show this, recall that $p_\psi(z) \propto \exp(-\psi(z))$ is the mixing distribution and $p(u|z) \propto \exp(-\frac{z}{2}u^2) \propto \mathcal{N}(u; 0, z^{-1})$ the mixture component distribution. Hence, the marginal distribution $p(u) \propto \exp(-\rho(u))$ is related to ρ as follows:

$$\rho'(u) = -\frac{d}{du} \log p(u) = -\frac{p'(u)}{p(u)} \tag{3.48}$$

$$p(u) = \int p_\psi(z) \mathcal{N}(u; 0, z^{-1}) \, dz \tag{3.49}$$

$$p'(u) = -u \int z p_\psi(z) \mathcal{N}(u; 0, z^{-1}) \, dz. \tag{3.50}$$

For MAP estimation, only the conditional distribution

$$p(z|u) = \frac{p_\psi(z)\mathcal{N}(u;0,z^{-1})}{\int p_\psi(z')\mathcal{N}(u;0,z^{-1})\,dz'} = \frac{p_\psi(z)\mathcal{N}(u;0,z^{-1})}{p(u)} \tag{3.51}$$

is important, since updating the latent variables depends on its expected value, which can be derived as:

$$\begin{aligned}\mathbb{E}[z|u] &= \frac{\int z p_\psi(z)\mathcal{N}(u;0,z^{-1})\,dz}{p(u)} \\ &= \frac{1}{p(u)}\frac{1}{-u}\left(-u\int z p_\psi(z)\mathcal{N}(u;0,z^{-1})\,dz\right) \\ &= -\frac{1}{u}\frac{p'(u)}{p(u)} = \frac{\rho'(u)}{u}.\end{aligned} \tag{3.52}$$

As a result, updating the latent variables z_{ic} line 4 of Alg. 3.3 is accomplished as

$$\mathbb{E}[z_{ic}|\mathbf{x},\mathbf{y}] = \frac{\rho_i'(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)})}{\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)}}, \tag{3.53}$$

which means that MAP estimation with the envelope and integral types are identical, since the update of the latent variables in line 4 of their respective algorithms is the same, and the algorithms do not differ otherwise (Fig. 3.3).

3.4.1.3 *Latent Gaussian MRF*

Whether we use the integral or envelope type, aim to do MAP estimation or sampling, in all cases we need to work with the distribution $p(\mathbf{x}|\mathbf{z},\mathbf{y})$, which given fixed latent variables $\mathbf{z}$ can be derived as the following multivariate Gaussian:

$$\begin{aligned}p(\mathbf{x}|\mathbf{z},\mathbf{y}) &\propto p(\mathbf{y}|\mathbf{x})p(\mathbf{x}|\mathbf{z}) \\ &\propto \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y}-\mathbf{K}\mathbf{x}\|^2 - \sum_{c\in\mathcal{C}}\sum_i \frac{z_{ic}}{2}(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)})^2\right) \\ &\propto \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y}-\mathbf{K}\mathbf{x}\|^2 - \frac{1}{2}\sum_i(\mathbf{F}_i\mathbf{x})^\mathsf{T}\mathbf{Z}_i(\mathbf{F}_i\mathbf{x})\right) \\ &\propto \exp\left(\mathbf{x}^\mathsf{T}\frac{\mathbf{K}^\mathsf{T}\mathbf{y}}{\sigma^2} - \frac{1}{2}\mathbf{x}^\mathsf{T}\frac{\mathbf{K}^\mathsf{T}\mathbf{K}}{\sigma^2}\mathbf{x} - \frac{1}{2}\mathbf{x}^\mathsf{T}\left(\sum_i \mathbf{F}_i^\mathsf{T}\mathbf{Z}_i\mathbf{F}_i\right)\mathbf{x}\right) \\ &\propto \exp\left(\mathbf{x}^\mathsf{T}\mathbf{K}^\mathsf{T}\mathbf{y}/\sigma^2 - \frac{1}{2}\mathbf{x}^\mathsf{T}\left(\mathbf{K}^\mathsf{T}\mathbf{K}/\sigma^2 + \sum_i \mathbf{F}_i^\mathsf{T}\mathbf{Z}_i\mathbf{F}_i\right)\mathbf{x}\right) \\ &\propto \mathcal{N}\left(\mathbf{x};\boldsymbol{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}}^{-1}\boldsymbol{\eta}_{\mathbf{x}|\mathbf{z},\mathbf{y}},\boldsymbol{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}}^{-1}\right)\end{aligned} \tag{3.54}$$

with

$$\boldsymbol{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}} = \mathbf{K}^\mathsf{T}\mathbf{K}/\sigma^2 + \sum_i \mathbf{F}_i^\mathsf{T}\mathbf{Z}_i\mathbf{F}_i \tag{3.55}$$

$$\boldsymbol{\eta}_{\mathbf{x}|\mathbf{z},\mathbf{y}} = \mathbf{K}^\mathsf{T}\mathbf{y}/\sigma^2, \tag{3.56}$$

$$\hat{z}_{ic}^{(t)} \leftarrow \rho_i'(\mathbf{f}_i^\mathsf{T}\hat{\mathbf{x}}_{(c)}^{(t-1)})/(\mathbf{f}_i^\mathsf{T}\hat{\mathbf{x}}_{(c)}^{(t-1)})$$

$$\hat{\mathbf{x}}^{(t)} \leftarrow \left(\mathbf{K}^\mathsf{T}\mathbf{K}/\sigma^2 + \sum_i \mathbf{F}_i^\mathsf{T}\,\mathcal{D}_\mathcal{C}\{\hat{z}_{ic}^{(t)}\}\mathbf{F}_i\right)^{-1}\left(\mathbf{K}^\mathsf{T}\mathbf{y}/\sigma^2\right)$$

Figure 3.3: Updates functions for MAP estimation (algorithms 3.1 and 3.3) with the multiplicative HQ form (envelope and integral type).

where $\mathbf{F}_i\mathbf{x} \equiv \mathbf{f}_i \otimes \mathbf{x} \equiv [\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(\mathcal{C}_1)}, \ldots, \mathbf{f}_i^\mathsf{T}\mathbf{x}_{(\mathcal{C}_{|\mathcal{C}|})}]^\mathsf{T}$ denotes convolution with filter $\mathbf{f}_i$ and $\mathbf{Z}_i = \mathcal{D}_\mathcal{C}\{z_{ic}\}$ is a diagonal matrix comprised of the elements z_{ic} for $c \in \mathcal{C}$.

Since $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$ is Gaussian, it is conceptually easy to compute its maximizer (used in line 5 of algorithms 3.1 and 3.3, *cf.* Fig. 3.3) or draw a sample from it (used in algorithms 3.2 and 3.4). Both can be accomplished by solving systems of linear equations based on the precision matrix $\mathbf{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}}$. However, since the precision matrix depends on the value of the latent variables $\mathbf{z}$, there are two computational disadvantages: *1)* Depending on the values of $\mathbf{z}$, the matrix might be poorly *conditioned*, which can lead to numerical instabilities or require a large number of iterations when using iterative equation system solvers (*cf.* Section 3.5). *2)* Direct equation system solvers use a factorization of the equation system matrix, which can be demanding in terms of computation and memory. Unfortunately, since updating the latent variables changes the system matrix, we have to re-compute such a factorization at every step of HQ inference. We will discuss these issues in more detail in Section 3.5.

3.4.1.4 *Example: Student-t potential*

We consider as a running example in this chapter the Student-t potential with (Lorentzian) penalty function

$$\rho(u) = \alpha \cdot \log\left(1 + \frac{1}{2}u^2\right), \tag{3.57}$$

for which we will derive half-quadratic representations to do MAP estimation.

ENVELOPE TYPE We start with the envelope type representation, where it is easy to show that $\rho(\sqrt{u})$ is a concave function. Thus, we can obtain $\psi(z)$ via Eq. (3.43) as

$$\begin{aligned} \psi(z) &= -\min_u \left\{\frac{1}{2}u^2 z - \alpha \log\left(1 + \frac{1}{2}u^2\right)\right\} \\ &= \begin{cases} \alpha \log(\alpha/z) + z - \alpha & z \leq \alpha \\ 0 & z > \alpha \end{cases} \end{aligned} \tag{3.58}$$

by finding the argument(s) where the derivative w.r.t. u is zero and then substituting back. Since there are multiple roots, we use the second derivative test to find the appropriate minima. Figure 3.2(a) visualizes the envelope type representation of the potential.

Next, we will compute Eq. (3.44) for our example, which provides us with the update function for the latent variables:

$$\arg\min_{z}\left\{\frac{1}{2}u^2 z+\psi(z)\right\}=\frac{\rho'(u)}{u}=\frac{\alpha}{1+\frac{1}{2}u^2}. \tag{3.59}$$

When a latent variable is chosen in this way, it induces a tight quadratic upper bound on the penalty function, which is shown in Fig. 3.4.

INTEGRAL TYPE For the integral type representation, we will derive the Student-t distribution as an infinite GSM, where it is known that the latent variable z is Gamma distributed:

Γ *denotes the gamma function.*

$$p_\psi(z)=g(z;a,b)\quad\text{with} \tag{3.60}$$

$$g(z;a,b)=\frac{1}{b^a\Gamma(a)}z^{a-1}e^{-z/b}. \tag{3.61}$$

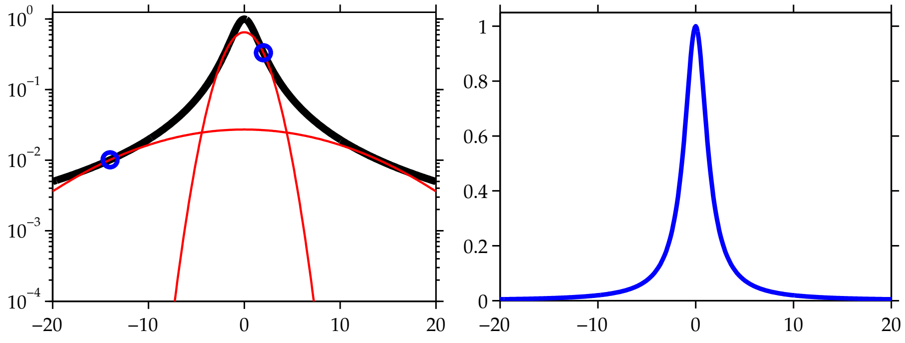
The latent variable determines the precision of the Gaussian mixture component $p(u|z)=\mathcal{N}(u;0,z^{-1})$. Hence, we obtain the joint distribution as

$$\begin{aligned}p(u,z)&=p_\psi(z)p(u|z)\\&=\frac{1}{b^a\Gamma(a)}z^{a-1}e^{-z/b}\sqrt{\frac{z}{2\pi}}\exp\left(-\frac{z}{2}u^2\right)\\&=\frac{1}{b^a\Gamma(a)\sqrt{2\pi}}z^{a-1/2}\exp\left(-z\left(\frac{1}{2}u^2+\frac{1}{b}\right)\right).\end{aligned} \tag{3.62}$$

We can verify that the marginal distribution

$$\begin{aligned}p(u)&=\int p(u,z)\,dz\\&=\frac{1}{b^a\Gamma(a)\sqrt{2\pi}}\int z^{a-1/2}\exp\left(-z\left(\frac{1}{2}u^2+\frac{1}{b}\right)\right)\,dz\\&=\frac{1}{b^a\Gamma(a)\sqrt{2\pi}}\Gamma\left(a+\frac{1}{2}\right)\left(\frac{1}{2}u^2+\frac{1}{b}\right)^{-a-1/2}\\&=\frac{\Gamma(\alpha)}{\Gamma(\alpha-\frac{1}{2})\sqrt{2\pi}}\left(1+\frac{1}{2}u^2\right)^{-\alpha}\\&\propto\exp(-\rho(u))\end{aligned} \tag{3.63}$$

does indeed yield the (normalized) Student-t potential for $b=1$ and $a=\alpha-1/2$. Here, we have used our knowledge about the integral $\int z^{a-1}e^{-z/b}\,dz=b^a\Gamma(a)$ from Eq. (3.61). Figure 3.2(b) visualizes the integral type representation of the potential.



(a) Lower bound

(b) $\rho'(u)/u$

Figure 3.4: **Lower bound for both types in multiplicative form (example).** *(a)* Student-t potential $\exp(-\rho(u))$ (thick black) with $\rho(u) = \log(1+\frac{1}{2}u^2)$ is tightly bounded for two selected values of u^* (blue circles) via $\exp(-\phi(u,z^*))$ with $z^* = \rho'(u^*)/u^*$ as shown in *(b)*.

The conditional distribution of the latent variables

$$\begin{aligned} p(z|u) &\propto p(u,z) \\ &\propto z^{a-1/2} \exp\left(-z\left(\frac{1}{2}u^2 + \frac{1}{b}\right)\right) \\ &\propto g\left(z; \alpha, \left(1+\frac{1}{2}u^2\right)^{-1}\right) \end{aligned} \tag{3.64}$$

is also Gamma distributed, where we have also substituted $b = 1$ and $a = \alpha - 1/2$ to match our setting. Updating the latent variables for EM (Alg. 3.3) requires computing the expected value of $p(z|u)$. The expected value of a Gamma distribution $g(z;a,b)$ is simply $a{\cdot}b$. Hence,

$$\mathbb{E}[z|u] = \alpha \cdot \left(1+\frac{1}{2}u^2\right)^{-1} = \frac{\alpha}{1+\frac{1}{2}u^2}, \tag{3.65}$$

which equals $\rho'(u)/u$ as shown earlier. Since the latent variables are chosen in the same way as in the envelope type (Eq. 3.59), they induce the same lower bound on the potential (Fig. 3.4).

3.4.1.5 *Connection with line processes and robust statistics*

Quadratic regularization has been used to impose smoothness between neighboring pixels (and other scene properties) in the early days of computer vision [*e.g.*, Horn and Schunck, 1981]. However, this started to change with the introduction of *line processes* [Geman and Geman, 1984] and also when ideas from *robust statistics* were applied to vision problems [Förstner, 1987].

ROBUST STATISTICS Modeling smoothness with quadratic penalty functions ρ assigns a high cost to outliers. However, as discussed earlier, such outliers can frequently occur at object boundaries in natural images. The idea of robust statistics is to deal with outliers that violate the model assumptions by replacing quadratic penalties with robust functions that assign a more moderate cost to outliers.

LINE PROCESS A line process [Geman and Geman, 1984] is a hierarchical graphical model, where unobserved variables **z** are introduced to denote the edges between all pairs of neighboring pixels, in addition to variables **x** that correspond to the pixels of the image. The idea is to explicitly model discontinuities in the images, apart from encouraging smoothness between neighboring pixels. This allows to model piece-wise smooth image regions, without enforcing global smoothness over the whole image. Mathematically, this may be formalized as adding an energy term

$$(x_i - x_j)^2 z_{ij} + \psi(z_{ij}) = \begin{cases} (x_i - x_j)^2 + \psi(1) & \text{if } z_{ij} = 1 \\ \psi(0) & \text{if } z_{ij} = 0 \end{cases} \tag{3.66}$$

with binary edge variables $z_{ij} \in \{0, 1\}$ for each pair of neighboring pixels x_i and x_j. This is called a *binary* line process since the edge variables z_{ij} only indicate the presence ($z_{ij} = 1$) or absence ($z_{ij} = 0$) of an edge between pixels x_i and x_j. The function $\psi(z_{ij})$ is basically chosen to specify a fixed cost $C > 0$ for not having an edge between two pixels, *e.g.* $\psi(0) = C$ and $\psi(1) = 0$. Additionally, Geman and Geman [1984] already proposed to also connect the variables **z** to model further properties, such as encouraging continuous edge segments. This corresponds to the notion of interacting auxiliary variables that we mentioned earlier.

Blake and Zisserman [1987] already showed that a (non-interacting) binary line process can be eliminated by minimizing over the edge variables for inference via energy minimization. Furthermore, the binary line process can be generalized into an *analog* line process by allowing each $z_{ij} \in \mathbb{R}^+$ to take on positive real values and extending the domain of ψ to $\mathbb{R}^+$. Hence, the multiplicative HQ form is an instance of an analog line process. Geman and Reynolds [1992] showed how an analog line process can be eliminated to yield a robust penalty function. A thorough treatment of the general equivalence between (analog) line processes and regularization with robust penalty functions is given by Black and Rangarajan [1996]. They provide a generic recipe to easily convert between robust penalties and their equivalent line process (*i. e.*, half-quadratic) representation.

3.4.1.6 *Connection with other optimization techniques*

LEAST SQUARES Idier [2001] discusses the connection between HQ inference and reweighted least squares approaches, which had been developed earlier in the signal processing community. Specifically, the multiplicative HQ form corresponds to the well-known method of *iteratively reweighted least squares* (IRLS) [*cf.* Rubin, 1983].

LINEAR GRADIENT APPROXIMATION Maximizing the posterior distribution $p(\mathbf{x}|\mathbf{y}) \propto \exp(-E(\mathbf{x}|\mathbf{y}))$ is equivalent to minimizing the associated energy $E(\mathbf{x}|\mathbf{y})$, whose gradient w.r.t. $\mathbf{x}$ can be written as

$$\begin{aligned} \nabla_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}) &= \nabla_{\mathbf{x}} \left[\frac{1}{2\sigma^2} \|\mathbf{y} - \mathbf{K}\mathbf{x}\|^2 + \sum_{c\in\mathcal{C}} \sum_i \rho_i(\mathbf{f}_{ic}^{\mathsf{T}}\mathbf{x}) \right] \\ &= \frac{1}{\sigma^2} \left(\mathbf{K}^{\mathsf{T}}\mathbf{K}\mathbf{x} - \mathbf{K}^{\mathsf{T}}\mathbf{y} \right) + \sum_{c\in\mathcal{C}} \sum_i \rho_i'(\mathbf{f}_{ic}^{\mathsf{T}}\mathbf{x}) \mathbf{f}_{ic} \cdot \frac{\mathbf{f}_{ic}^{\mathsf{T}}\mathbf{x}}{\mathbf{f}_{ic}^{\mathsf{T}}\mathbf{x}} \\ &= \left(\frac{1}{\sigma^2} \mathbf{K}^{\mathsf{T}}\mathbf{K} \right) \mathbf{x} + \left(\sum_{c\in\mathcal{C}} \sum_i \mathbf{f}_{ic} \frac{\rho_i'(\mathbf{f}_{ic}^{\mathsf{T}}\mathbf{x})}{\mathbf{f}_{ic}^{\mathsf{T}}\mathbf{x}} \mathbf{f}_{ic}^{\mathsf{T}} \right) \mathbf{x} - \frac{1}{\sigma^2} \mathbf{K}^{\mathsf{T}}\mathbf{y} \\ &= \mathbf{A}(\mathbf{x})\mathbf{x} - \mathbf{b}, \end{aligned} \tag{3.67}$$

with

$$\mathbf{A}(\mathbf{x}) = \mathbf{K}^{\mathsf{T}}\mathbf{K}/\sigma^2 + \sum_i \mathbf{F}_i^{\mathsf{T}} \mathcal{D}_{\mathcal{C}} \{ \rho_i'(\mathbf{f}_{ic}^{\mathsf{T}}\mathbf{x}) / (\mathbf{f}_{ic}^{\mathsf{T}}\mathbf{x}) \} \mathbf{F}_i \tag{3.68}$$

$$\mathbf{b} = \mathbf{K}^{\mathsf{T}}\mathbf{y}/\sigma^2 \tag{3.69}$$

and $\mathbf{F}_i\mathbf{x} \equiv \mathbf{f}_i \otimes \mathbf{x}$ denoting convolution as before; $\mathcal{D}_{\mathcal{C}}\{.\}$ is a diagonal matrix with entries for $c \in \mathcal{C}$.

A (local) optimum of the energy function can be characterized by $\nabla_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}) = \mathbf{A}(\mathbf{x})\mathbf{x} - \mathbf{b} = \mathbf{0}$, but directly solving for $\mathbf{x}$ is generally intractable. However, we can devise an iterative algorithm that uses a gradient approximation around the current solution, where we can solve this. When we linearly approximate the gradient around $\hat{\mathbf{x}}^{(t-1)}$ as $\nabla_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}) \approx \mathbf{A}\big(\hat{\mathbf{x}}^{(t-1)}\big)\mathbf{x} - \mathbf{b}$, we can iteratively update the solution as follows:

$$\begin{aligned} \hat{\mathbf{x}}^{(t)} &\leftarrow \mathbf{A}\big(\hat{\mathbf{x}}^{(t-1)}\big)^{-1}\mathbf{b} \\ &= \left(\mathbf{K}^{\mathsf{T}}\mathbf{K}/\sigma^2 + \sum_i \mathbf{F}_i^{\mathsf{T}} \mathcal{D}_{\mathcal{C}} \left\{ \frac{\rho_i'(\mathbf{f}_{ic}^{\mathsf{T}}\hat{\mathbf{x}}^{(t-1)})}{\mathbf{f}_{ic}^{\mathsf{T}}\hat{\mathbf{x}}^{(t-1)}} \right\} \mathbf{F}_i \right)^{-1} \left(\mathbf{K}^{\mathsf{T}}\mathbf{y}/\sigma^2 \right). \end{aligned} \tag{3.70}$$

Note that this corresponds exactly to the update step in the multiplicative HQ form, as summarized in Fig. 3.3. The general equivalence between MAP estimation in the multiplicative HQ form and this iterative gradient linearization approach (as shown above) has first been shown by Nikolova and Chan [2007]. They further demonstrate the correspondence to a *quasi-Newton* approach [*cf.* Nocedal and Wright, 1999, § 8]

$$\begin{aligned} \hat{\mathbf{x}}^{(t)} &\leftarrow \mathbf{A}\big(\hat{\mathbf{x}}^{(t-1)}\big)^{-1}\mathbf{b} \\ &= \mathbf{A}\big(\hat{\mathbf{x}}^{(t-1)}\big)^{-1} \Big(\mathbf{A}\big(\hat{\mathbf{x}}^{(t-1)}\big)\hat{\mathbf{x}}^{(t-1)} - \mathbf{A}\big(\hat{\mathbf{x}}^{(t-1)}\big)\hat{\mathbf{x}}^{(t-1)} + \mathbf{b} \Big) \\ &= \hat{\mathbf{x}}^{(t-1)} - \mathbf{A}\big(\hat{\mathbf{x}}^{(t-1)}\big)^{-1} \nabla_{\hat{\mathbf{x}}^{(t-1)}} E\big(\hat{\mathbf{x}}^{(t-1)}|\mathbf{y}\big), \end{aligned} \tag{3.71}$$

where the *Hessian* matrix of $E(\mathbf{x}|\mathbf{y})$ is approximated by $\mathbf{A}(\hat{\mathbf{x}}^{(t-1)})$; this had previously been discussed for convex penalty functions [Nikolova and Ng, 2005; Allain et al., 2006]. Furthermore, Nikolova and Chan [2007] show that this iterative algorithm can be seen as an instance of the *generalized Weiszfeld's algorithm*, which has also been discussed by Chan and Mulet [1999] and Allain et al. [2006].

Weiszfeld's algorithm *for finding the point with the closest Euclidean distance to a set of other points dates back to Weiszfeld [1937].*

It may seem surprising that the HQ approach is equivalent to a simple linearization of the energy's gradient. However, since the HQ approximation of the energy is quadratic, it directly follows that this must correspond to a particular linear approximation of the energy's gradient. Also note that we basically did not make any assumptions when deriving the gradient linearization, which yields an *approximation* of the energy. In contrast, the HQ formulation guarantees an upper *bound* on the energy. As a consequence, we do not need to worry about step-sizes in the iterative HQ energy minimization algorithm, since we minimize a quadratic upper bound at each step. On the other hand, we can still apply the iterative gradient linearization approach even when the penalty functions ρ_i do not have a HQ representation; however, we should find an appropriate step-size in this case, since the quadratic approximation is not a lower bound of the energy.

3.4.2 *Additive form*

The additive half-quadratic form has arguably received less attention in the literature, presumably because it is less intuitive and cannot directly be applied to popular non-smooth potentials, such as (hyper-)Laplacians, as we will show later.

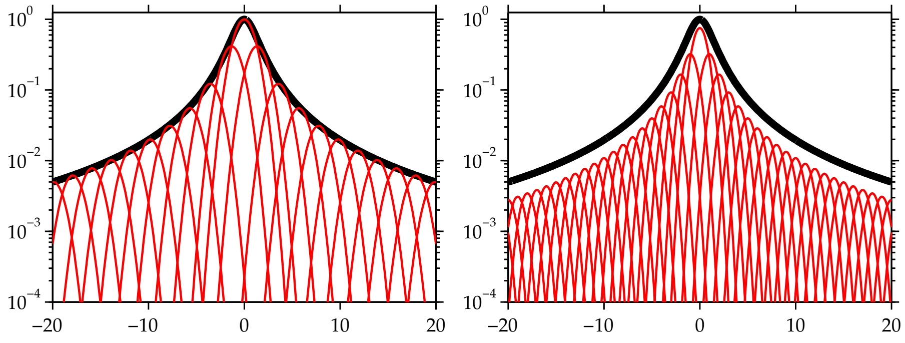
3.4.2.1 *Envelope type*

We will again first address the envelope type, where (with $\bigoplus = \max$) we need to satisfy the equation

$$\rho(u) = \min_z \phi(u,z) = \min_z \left\{ \frac{\beta}{2}(u-z)^2 + \psi(z) \right\} \tag{3.72}$$

by choosing a suitable function ψ and scaling factor $\beta > 0$. We start by transforming the equation in order to use convex duality properties:

$$\begin{aligned}
& & \rho(u) &= \min_z \left\{ \frac{\beta}{2}(u-z)^2 + \psi(z) \right\} & (3.73)\\
&\Leftrightarrow & \rho(u) - \frac{\beta}{2}u^2 &= \min_z \left\{ -\beta \cdot uz + \psi(z) + \frac{\beta}{2}z^2 \right\} & (3.74)\\
&\Leftrightarrow & \rho(-v/\beta) - \frac{1}{2\beta}v^2 &= \min_z \left\{ vz - \left(-\psi(z) - \frac{\beta}{2}z^2 \right) \right\} & (3.75)\\
&\Leftrightarrow & f(v) &= \min_z \left\{ vz - f^*(z) \right\}. & (3.76)
\end{aligned}$$



(a) Envelope type (b) Integral type

Figure 3.5: **Additive form (example).** HQ representation of a Student-t potential $\exp(-\rho(u))$ (thick black) with $\rho(u) = \log(1+\frac{1}{2}u^2)$ and the associated $\exp(-\phi(u,z))$ for a few values of z (red). Envelope type representation in *(a)* is exact with $\beta = \alpha = 1$, whereas integral type representation in *(b)* is an approximation (finite GLM using scaling $\beta = 2$); *(b)* is also scaled by for better comparison.

We have used the variable transformation $u = -v/\beta$ and obtained $f(v) = \rho(-v/\beta) - \frac{1}{2\beta}v^2$ and its dual $f^*(z) = -\psi(z) - \frac{\beta}{2}z^2$. Hence, we can use the additive form for a given ρ if there exists a value $\beta > 0$ such that $f(v)$ is a concave function. For instance, if ρ is twice differentiable, we can show concavity of f by verifying that there is a $\beta > 0$ such that $f''(v) \leq 0$ for all $v \in \mathbb{R}$. Note that in practice, we want to find the smallest such β, since it intuitively will lead to the broadest possible quadratic function and as such a better bound on the potential (*cf*. Fig. 3.7(a)). If the additive form is applicable to ρ, we can obtain ψ as

$$\psi(z) = -\left(f^*(z) + \frac{\beta}{2}z^2\right) \tag{3.77}$$

$$= -\min_v \left\{ vz - \rho(-v/\beta) + \frac{1}{2\beta}v^2 + \frac{\beta}{2}z^2 \right\} \tag{3.78}$$

$$= -\min_{-u\beta} \left\{ -\beta \cdot uz - \rho(u) + \frac{\beta}{2}u^2 + \frac{\beta}{2}z^2 \right\} \tag{3.79}$$

$$= -\min_u \left\{ \frac{\beta}{2}(u-z)^2 - \rho(u) \right\} \tag{3.80}$$

where we have substituted back $v = -u\beta$.

MAP ESTIMATION Unfortunately, especially in the additive form, an analytical expression for ψ is often difficult. Luckily, updating the latent variables for a given value of u for MAP estimation is also easy in the additive form, where we again use the properties of convex duality. First, it is easy to convince ourselves that the minimizer of the right-hand side (RHS) of Eq. (3.76) also minimizes the RHS of Eq. (3.73). Thus,

$$\arg\min_z \left\{ \frac{\beta}{2}(u-z)^2 + \psi(z) \right\} = -\frac{d\rho(v)}{dv} \cdot \frac{1}{\beta} - \frac{v}{\beta} = u - \frac{\rho'(u)}{\beta} \tag{3.81}$$

where we have substituted back $v = -u\beta$ and used that $f'(v) = \arg\min_z \{vz - f^*(z)\}$. Hence, updating the latent variables z_{ic} during half-quadratic MAP estimation, *i. e.* line 4 of Alg. 3.1 can be carried out as follows (*cf.* Fig. 3.6):

$$\arg\max_{z_{ic}} p(z_{ic}|\mathbf{x},\mathbf{y}) = \mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)} - \frac{\rho_i'(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)})}{\beta}. \tag{3.82}$$

3.4.2.2 *Integral type*

In the integral type, we again choose $\bigoplus = \int$, hence

$$\begin{aligned} \exp(-\rho(u)) &= \int \exp(-\phi(u,z))\; dz \\ &= \int \exp(-\psi(z)) \cdot \exp\Big(-\frac{\beta}{2}(u-z)^2\Big)\; dz \\ &\propto \int p_\psi(z) \cdot \mathcal{N}\big(u; z, \beta^{-1}\big)\; dz, \end{aligned} \tag{3.83}$$

where $p_\psi(z)$ is the mixing distribution of the latent variable z that now determines the mean of the Gaussian mixture component with fixed variance β^{-1}. Equation (3.83) is also known as the model class of Gaussian location mixtures (GLMs) or *Gaussian convolutions* [*e. g.*, DasGupta, 1994].

Polson and Scott [2016] study the conditions under which a potential function has both an envelope and integral type representation in the additive form. While we do not re-state the existence conditions for the integral type here, they are rather restrictive (compared to the multiplicative form) and only partially overlap with the necessary conditions for the envelope type (as stated earlier). DasGupta [1994] characterizes all GSMs that can also be represented as GLMs, and Polson and Scott [2016] remark that this rules out a GLM representation for many commonly used distributions, such as the Student-t, which are well-known to be GSMs.

As in the multiplicative form, we are not aware of a simple recipe to convert a suitable potential into a GLM, which is again in contrast to the envelope type where convex duality properties can be used to derive the HQ representation. However, as in the multiplicative form, choosing a mixing distribution $p_\psi(z)$ and then obtaining the associated GLM potential can also be applied here.

MAP ESTIMATION Fortunately, MAP estimation is also not predicated on finding a GLM representation for the potential, since the

$$\hat{z}_{ic}^{(t)} \leftarrow \mathbf{f}_i^\mathsf{T} \hat{\mathbf{x}}_{(c)}^{(t-1)} - \rho_i'(\mathbf{f}_i^\mathsf{T} \hat{\mathbf{x}}_{(c)}^{(t-1)})/\beta$$

$$\hat{\mathbf{x}}^{(t)} \leftarrow \left(\mathbf{K}^\mathsf{T}\mathbf{K}/\sigma^2 + \beta \sum_i \mathbf{F}_i^\mathsf{T}\mathbf{F}_i\right)^{-1} \left(\mathbf{K}^\mathsf{T}\mathbf{y}/\sigma^2 + \beta \sum_i \mathbf{F}_i^\mathsf{T} \operatorname{vec}_\mathcal{C}\{\hat{z}_{ic}^{(t)}\}\right)$$

Figure 3.6: Updates functions for MAP estimation (algorithms 3.1 and 3.3) with the additive HQ form (envelope and integral type).

update of the latent variables only depends on the penalty ρ. Let us first relate the marginal distribution $p(u)$ to the penalty function ρ as follows:

$$\rho'(u) = -\frac{d}{du}\log p(u) = -\frac{p'(u)}{p(u)} \tag{3.84}$$

$$p(u) = \int p_\psi(z)\mathcal{N}(u; z, \beta^{-1})\ dz \tag{3.85}$$

$$p'(u) = -\beta \int (u-z) p_\psi(z)\mathcal{N}(u; z, \beta^{-1})\ dz. \tag{3.86}$$

Again, only the conditional distribution

$$p(z|u) = \frac{p_\psi(z)\mathcal{N}(u; z, \beta^{-1})}{\int p_\psi(z')\mathcal{N}(u; z', \beta^{-1})\ dz'} = \frac{p_\psi(z)\mathcal{N}(u; z, \beta^{-1})}{p(u)} \tag{3.87}$$

is necessary for updating the latent variables. The update equation is based on the expected value (Alg. 3.3), which is obtained as

$$\mathbb{E}[z|u] = \frac{\int z p_\psi(z)\mathcal{N}(u; z, \beta^{-1})\ dz}{p(u)} \tag{3.88}$$

$$= \frac{-\beta\int (u-z) p_\psi(z)\mathcal{N}(u; z, \beta^{-1})\ dz + \beta u p(u)}{\beta p(u)} \tag{3.89}$$

$$= \frac{p'(u) + \beta u p(u)}{\beta p(u)} \tag{3.90}$$

$$= u - \frac{\rho'(u)}{\beta}. \tag{3.91}$$

Hence, during MAP estimation each latent variable z_{ic} (line 4 of Alg. 3.3) is updated as

$$\mathbb{E}[z_{ic}|\mathbf{x}, \mathbf{y}] = \mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)} - \frac{\rho_i'(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)})}{\beta}. \tag{3.92}$$

As a result, MAP estimation with the envelope and integral types are also identical for the additive form, since the update of the latent variables in line 4 of their respective algorithms is the same, and the algorithms do not differ otherwise (Fig. 3.6).

3.4.2.3 *Latent Gaussian MRF*

As in the multiplicative form, the latent variables determine a Gaussian MRF $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$ that is used to update the image for HQ inference for both integral or envelope type. The form of the multivariate Gaussian is given as

$$
\begin{aligned}
p(\mathbf{x}|\mathbf{z},\mathbf{y}) &\propto p(\mathbf{y}|\mathbf{x})p(\mathbf{x}|\mathbf{z}) \\
&\propto \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y}-\mathbf{K}\mathbf{x}\|^2 - \sum_{c\in\mathcal{C}}\sum_i \frac{\beta}{2}(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)} - z_{ic})^2\right) \\
&\propto \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y}-\mathbf{K}\mathbf{x}\|^2 - \frac{\beta}{2}\sum_i(\mathbf{F}_i\mathbf{x}-\mathbf{z}_i)^\mathsf{T}(\mathbf{F}_i\mathbf{x}-\mathbf{z}_i)\right) \\
&\propto \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{y}-\mathbf{K}\mathbf{x}\|^2 + \mathbf{x}^\mathsf{T}\beta\sum_i \mathbf{F}_i^\mathsf{T}\mathbf{z}_i - \frac{1}{2}\mathbf{x}^\mathsf{T}\left(\beta\sum_i \mathbf{F}_i^\mathsf{T}\mathbf{F}_i\right)\mathbf{x}\right) \\
&\propto \exp\left(\mathbf{x}^\mathsf{T}\left(\frac{\mathbf{K}^\mathsf{T}\mathbf{y}}{\sigma^2} + \beta\sum_i \mathbf{F}_i^\mathsf{T}\mathbf{z}_i\right) - \frac{1}{2}\mathbf{x}^\mathsf{T}\left(\frac{\mathbf{K}^\mathsf{T}\mathbf{K}}{\sigma^2} + \beta\sum_i \mathbf{F}_i^\mathsf{T}\mathbf{F}_i\right)\mathbf{x}\right) \\
&\propto \mathcal{N}\left(\mathbf{x};\boldsymbol{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}}^{-1}\boldsymbol{\eta}_{\mathbf{x}|\mathbf{z},\mathbf{y}},\boldsymbol{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}}^{-1}\right) \qquad (3.93)
\end{aligned}
$$

with

$$
\boldsymbol{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}} = \mathbf{K}^\mathsf{T}\mathbf{K}/\sigma^2 + \beta\sum_i \mathbf{F}_i^\mathsf{T}\mathbf{F}_i \qquad (3.94)
$$

$$
\boldsymbol{\eta}_{\mathbf{x}|\mathbf{z},\mathbf{y}} = \mathbf{K}^\mathsf{T}\mathbf{y}/\sigma^2 + \beta\sum_i \mathbf{F}_i^\mathsf{T}\mathbf{z}_i \qquad (3.95)
$$

where $\mathbf{F}_i\mathbf{x} \equiv \mathbf{f}_i \otimes \mathbf{x} \equiv [\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(\mathcal{C}_1)},\ldots,\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(\mathcal{C}_{|\mathcal{C}|})}]^\mathsf{T}$ as usual denotes convolution with filter $\mathbf{f}_i$ and $\mathbf{z}_i = \mathrm{vec}_{\mathcal{C}}\{z_{ic}\}$ is a vector of all z_{ic} for $c \in \mathcal{C}$.

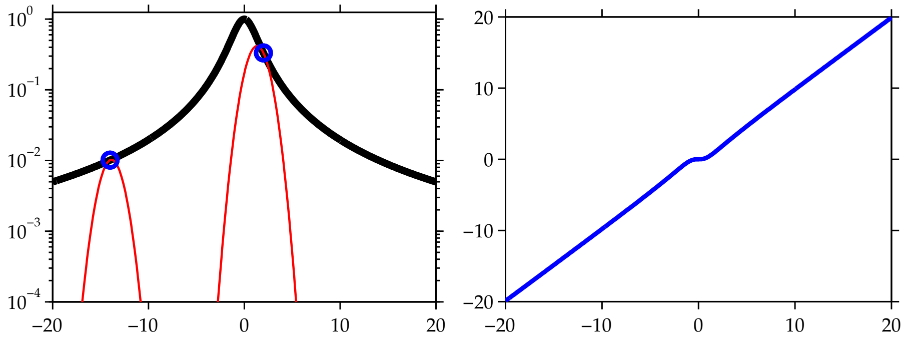
Again, since $p(\mathbf{x}|\mathbf{z},\mathbf{y})$ is Gaussian, finding its mode (Fig. 3.6, line 5 of algorithms 3.1 and 3.3) or drawing a sample (algorithms 3.2 and 3.4) is conceptually easy based on solving systems of linear equations with precision matrix $\boldsymbol{\Omega}_{\mathbf{x}|\mathbf{z},\mathbf{y}}$. In contrast to the multiplicative form, the precision matrix does not depend on the value of the latent variables $\mathbf{z}$, which has computational advantages. The matrix has homogeneous structure, which may allow to apply special decomposition techniques; furthermore, the matrix might also be better conditioned. In addition, we can re-use a matrix factorization for all iterations of HQ inference, which can save a lot of computation time. Again, further details will be discussed in Section 3.5.

3.4.2.4 *Example: Student-t potential*

We return to our example of the Student-t distribution, which we will now represent in the additive half-quadratic form.

ENVELOPE TYPE In the envelope type, we first need to check if $\rho(u) = \alpha\log(1+\frac{1}{2}u^2)$ is actually representable in the additive form. To that end, as stated earlier, we need to find $\beta > 0$ such that

$$
f(v) = \rho(-v/\beta) - \frac{1}{2\beta}v^2 = \alpha\log\left(1+\frac{1}{2\beta^2}v^2\right) - \frac{1}{2\beta}v^2 \qquad (3.96)
$$



(a) Lower bound

(b) $u - \rho'(u)/\alpha$

Figure 3.7: **Lower bound for both types in additive form (example).** *(a)* Student-t potential $\exp(-\rho(u))$ (thick black) with $\rho(u) = \log(1 + \frac{1}{2}u^2)$ is tightly bounded for two selected values of u^* (blue circles) via $\exp(-\phi(u, z^*))$ with $z^* = u^* - \rho'(u^*)/\alpha$ as shown in *(b)*.

is a concave function of v, which is the case if

$$f''(v) = \frac{\alpha(4\beta^3 - 2\beta v^2)}{\beta(2\beta^2 + v^2)^2} - \frac{1}{\beta} \le 0 \tag{3.97}$$

for $v \in \mathbb{R}$. Assuming $\alpha > 0$, the above inequality is only satisfied when $\beta \ge \alpha$. As mentioned earlier, we want to find the smallest β that fulfills the conditions, hence we choose $\beta = \alpha$.

We could go on and compute $\psi(z) = -(f^*(z) + \frac{\alpha}{2}z^2)$, but it does not have a simple analytical expression. Hence, we will only compute it numerically for illustration purposes in Figs. 3.5 and 3.7. However, computing the update of the latent variables is easy as

$$\arg\min_z \left\{ \frac{\alpha}{2}(u - z)^2 + \psi(z) \right\} = u - \frac{\rho'(u)}{\alpha} = u - \frac{u}{1 + \frac{1}{2}u^2}, \tag{3.98}$$

which is shown in Fig. 3.7(b).

INTEGRAL TYPE Since an exact HQ representation for the integral type via an infinite GLM seems not possible, we have approximated the potential function with a finite GLM, which is shown in Fig. 3.5(b). In particular, we fit the mixture weights π_z of a multinomial distribution with $p_\psi(z) = \pi_z$, such that $\sum_z \pi_z \mathcal{N}(u; z, \beta^{-1})$ closely matches the Student-t distribution. If an exact HQ representation were possible, half-quadratic MAP estimation would be exactly the same as in the envelope type.

3.4.2.5 *Connection with proximal and constrained optimization methods*

The additive HQ form can be interpreted in the context of *proximal* algorithms [*cf.* Parikh and Boyd, 2013], which are typically used to solve convex, but often non-smooth or large-scale, optimization problems. A connection between half-quadratic inference and proximal methods has been made by Polson and Scott [2016], who specifically show a link to the *proximal gradient* algorithm [*cf.* Parikh and Boyd, 2013, § 4.2].

PROXIMAL INTERPRETATION Given a function f, its *Moreau envelope* with parameter $\lambda > 0$ is typically defined as

$$M_{\lambda f}(u) = \min_z \left\{ \frac{1}{2\lambda}(u-z)^2 + f(z) \right\}, \tag{3.99}$$

which can be interpreted as a smoothed or regularized approximation of f. Furthermore, the unique minimizer of Eq. (3.99) is called the *proximal operator*

$$\operatorname{prox}_{\lambda f}(u) = \arg\min_z \left\{ \frac{1}{2\lambda}(u-z)^2 + f(z) \right\} \tag{3.100}$$

of function f with parameter $\lambda > 0$. Intuitively, $\operatorname{prox}_{\lambda f}(u)$ strikes a balance (weighted by λ) between minimizing f, but at the same time staying close to u. Depending on the context and the particular function f, Eq. (3.100) is also known as *shrinkage function* [*e.g.*, Donoho, 1995; Beck and Teboulle, 2009].

Given these definitions, we can interpret the (envelope type) additive HQ representation of Eq. (3.72) as follows. Applying Eq. (3.99) with $\lambda = 1/\beta$, the penalty function ρ is the Moreau envelope of the auxiliary function ψ. This intuitively explains why the additive HQ form does not exist for non-smooth ρ (such as $\rho(u) = |u|$, see below), since ρ itself is a smoothed variant of ψ. Furthermore, the update equation for the latent variables in Eq. (3.81) is the proximal operator of auxiliary function ψ.

LAPLACIAN EXAMPLE While the Moreau envelope interpretation intuitively explains why the additive HQ form is not applicable to the popular, but non-smooth, Laplacian potential with L_1 penalty function $\rho(u) = |u|$, we can also show this formally, using the criteria introduced earlier. Specifically, if the additive form can be applied, then a value $\beta > 0$ must exist such that

$$f(v) = \rho(-v/\beta) - v^2/(2\beta) = \frac{|v| - v^2/2}{\beta} \tag{3.101}$$

is a concave function (*cf.* Eq. 3.76). However, we can verify that the definition of concavity

$$f\big((1-t)a + (t)b\big) \geq (1-t)f(a) + (t)f(b), \tag{3.102}$$

which must hold for all $a, b \in \mathbb{R}$ and $t \in [0,1]$, is violated for $a = -1, b = 1, t = 1/2$:

$$0 = f(0) \ngeq \frac{f(-1) + f(1)}{2} = \frac{1}{2\beta}. \tag{3.103}$$

Hence, a HQ representation of $\rho(u) = |u|$ in the additive form (Eq. 3.72) is only applicable in the limit $\beta \to \infty$ if concavity is satisfied otherwise (which can be shown). When β goes to infinity, the quadratic upper bound on ρ induced by the additive form visually looks more and more peaked and eventually degenerates to a *Dirac delta* function, which can obviously bound any function perfectly. However, bounding the penalty function locally with a Dirac delta is not useful for optimization, since the alternating inference algorithm would simply "get stuck".

PENALTY METHOD Nevertheless, considering what happens when $\beta \to \infty$ has its merits. Recall that ρ can be seen as a smoothed version of ψ, where β in fact controls the amount of smoothness. Since smoothing decreases as β gets larger, ρ and ψ in fact become the same function in the limit of $\beta \to \infty$. Hence, we do not need ψ anymore, because we can replace it by ρ and relate the penalty function to itself via

$$\rho(u) \equiv \lim_{\beta\to\infty} \min_{z} \left\{ \frac{\beta}{2}(u-z)^2 + \rho(u) \right\}, \tag{3.104}$$

since the minimum on the RHS is achieved at $z = u$, since all other values of z lead to an infinite cost. While this may look trivial, it is part of the well-known *penalty method* [*cf.* Nocedal and Wright, 1999, § 17] for *constrained* optimization problems, here with an equality constraint between u and z.

To see why this is useful and how it applies to our setting, let us start from the beginning and consider the original posterior distribution that we want to maximize:

$$p(\mathbf{x}|\mathbf{y}) \propto \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2\mathbf{I}) \cdot \prod_{c\in\mathcal{C}} \prod_{i=1}^{N} \exp\left(-\rho_i(\mathbf{f}_i^{\mathsf{T}}\mathbf{x}_{(c)})\right). \tag{3.105}$$

By using *variable splitting*, *i.e.* introducing auxiliary variables z_{ic} and imposing equality constraints, we can formulate the MAP solution as the following equivalent constrained optimization problem:

$$\begin{aligned} \max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y}) &= \max_{\mathbf{x},\mathbf{z}} \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2\mathbf{I}) \cdot \prod_{c\in\mathcal{C}} \prod_{i=1}^{N} \exp\left(-\rho_i(z_{ic})\right) \\ \text{subject to} \quad & z_{ic} = \mathbf{f}_i^{\mathsf{T}}\mathbf{x}_{(c)}, \quad c \in \mathcal{C},\ i = 1, \ldots, N. \end{aligned} \tag{3.106}$$

Applying a (quadratic) penalty method, we can transform the above constrained problem into an easier to solve unconstrained one by adding terms that penalize violations of the equality constraints:

$$\max_{\mathbf{x},\mathbf{z}} \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2\mathbf{I}) \cdot \prod_{c\in\mathcal{C}} \prod_{i=1}^{N} \exp\left(-\frac{\beta}{2}(\mathbf{f}_i^{\mathsf{T}}\mathbf{x}_{(c)} - z_{ic})^2 - \rho_i(z_{ic})\right). \tag{3.107}$$

The optimization then alternates between solving the unconstrained problem w.r.t. $\mathbf{x}$ and $\mathbf{z}$, while the penalty parameter β must be increased after every step, such that $\beta \to \infty$ to eventually ensure solution of the original problem in Eq. (3.106). Of course, such a continuation scheme is in practice only applied until a desired accuracy is reached. The alternating optimization of Eq. (3.107) closely resembles MAP estimation in the additive form, with the difference that ρ is used instead of ψ and that β is gradually increased. Such a penalty-based approach has been taken by Wang et al. [2008], who also put their method in context of the additive HQ approach. Building upon the work of Wang et al. [2008], Krishnan and Fergus [2009] popularized this approach for the application of non-blind deconvolution with hyper-Laplacian potential functions.

FURTHER GENERALIZATION Assuming $\rho_i = \rho$, we can write the MAP solution of Eq. (3.105) and thus Eq. (3.106) after splitting variables more compactly as

$$\min_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{A}\mathbf{x}) \;=\; \min_{\mathbf{x},\mathbf{z}} f(\mathbf{x}) + g(\mathbf{z}) \quad \text{s.t.} \quad \mathbf{z} = \mathbf{A}\mathbf{x}, \tag{3.108}$$

with $f(\mathbf{x}) = \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{K}\mathbf{x}\|^2$, $g(\mathbf{z}) = \sum_j \rho(z_j)$, and $\mathbf{A} = [\mathbf{F}_1^\mathsf{T}, \ldots, \mathbf{F}_N^\mathsf{T}]^\mathsf{T}$, where $\mathbf{F}_i$ denotes convolution with $\mathbf{f}_i$ as before. The quadratic penalty formulation of Eq. (3.107) can now concisely be expressed as

$$\min_{\mathbf{x},\mathbf{z}} f(\mathbf{x}) + g(\mathbf{z}) + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{z}\|^2, \tag{3.109}$$

which necessitates to increase $\beta \to \infty$ to ensure solution of the original problem, which is sub-optimal numerically and w.r.t. convergence [Nocedal and Wright, 1999, § 17.1].

To avoid this issue, an alternative approach is known as *augmented Lagrangian method* [*cf.* Nocedal and Wright, 1999, § 17.4], which applied to our setting optimizes

$$\min_{\mathbf{x},\mathbf{z}} f(\mathbf{x}) + g(\mathbf{z}) + \frac{\beta}{2}\|\mathbf{A}\mathbf{x} - \mathbf{z}\|^2 - \mathbf{w}^\mathsf{T}(\mathbf{A}\mathbf{x} - \mathbf{z}), \tag{3.110}$$

with an estimate of *Lagrange multipliers* $\mathbf{w}$ to make up for the error in the approximation when $\beta < \infty$. Hence, it may no longer be necessary to increase β to large values, and thus avoid numerical problems. Minimization is carried out w.r.t. $\mathbf{x}$ and $\mathbf{z}$ in each iteration as before, but additionally includes the update step $\mathbf{w} \leftarrow \mathbf{w} - \beta(\mathbf{A}\mathbf{x} - \mathbf{z})$ to improve the estimate of the Lagrange multipliers.

The augmented Lagrangian method is also known as *method of multipliers*. The variant used here is typically called *alternating direction method of multipliers* (ADMM), since we do not jointly optimize Eq. (3.110) w.r.t. $\{\mathbf{x}, \mathbf{z}\}$, but instead alternate between minimizing w.r.t. $\mathbf{x}$ and $\mathbf{z}$ in each iteration. ADMM is an instance of *Douglas-Rachford splitting* and can be seen as a proximal algorithm [*cf.* Parikh and Boyd, 2013, § 4.4]. A widely-applicable proximal method for convex problems in the context of imaging is due to Chambolle and Pock [2011], which can be understood as a preconditioned version of ADMM.

CONVEX OPTIMIZATION In general, Eq. (3.108) and similar optimization problems are typically only addressed in the optimization literature under the assumption of convexity (here, functions f and g); a comprehensive overview of many proximal splitting methods for signal and image processing applications is given by Combettes and Pesquet [2011]. Note that L_1 *regularization*, *i. e.* $\rho = |.| \Rightarrow g(\mathbf{z}) = \|\mathbf{z}\|_1$, is commonly used, since it favors the most sparse solution while still leading to a convex optimization problem. For image processing problems, *total variation* (TV) regularization [Rudin et al., 1992] is particularly popular, which can roughly be seen as a special case of our setting with $g(\mathbf{z}) = \|\mathbf{z}\|_1$ and first-order derivative filters $\mathbf{f}_1 = [1, -1]^\mathsf{T}, \mathbf{f}_2 = [1, -1]$ (*anisotropic* TV). For instance, using TV regularization with the data term $f(\mathbf{x}) = \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{x}\|^2$ is well-known in the literature as *Rudin-Osher-Fatemi* denoising [Rudin et al., 1992]. Many algorithms have specifically been introduced to address TV/L_1 regularization; for instance, an effective variant of ADMM tailored to L_1 regularization has been proposed by Goldstein and Osher [2009] under the name *Split Bregman method*.

3.4.2.6 *Connection with other optimization techniques*

LINEAR GRADIENT APPROXIMATION For the multiplicative HQ form, Nikolova and Chan [2007] have shown the equivalence to an iterative gradient linearization algorithm (*cf.* Section 3.4.1). However, they have not explicitly discussed an analog result for the additive form, although it is strongly implied by earlier work [Nikolova and Ng, 2005; Allain et al., 2006]. Nevertheless, to remedy this we again start by bringing the gradient of energy $E(\mathbf{x}|\mathbf{y})$ into a particular form, here as

$$\begin{aligned}
\nabla_\mathbf{x} E(\mathbf{x}|\mathbf{y}) &= \nabla_\mathbf{x}\left[\frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{K}\mathbf{x}\|^2 + \sum_{c\in\mathcal{C}}\sum_i \rho_i(\mathbf{f}_{ic}^\mathsf{T}\mathbf{x})\right] \\
&= \frac{1}{\sigma^2}\left(\mathbf{K}^\mathsf{T}\mathbf{K}\mathbf{x} - \mathbf{K}^\mathsf{T}\mathbf{y}\right) + \sum_{c\in\mathcal{C}}\sum_i \rho_i'(\mathbf{f}_{ic}^\mathsf{T}\mathbf{x})\mathbf{f}_{ic} + (1-1)\beta\mathbf{f}_{ic}\mathbf{f}_{ic}^\mathsf{T}\mathbf{x} \\
&= \left(\frac{\mathbf{K}^\mathsf{T}\mathbf{K}}{\sigma^2} + \beta\sum_{c\in\mathcal{C}}\sum_i \mathbf{f}_{ic}\mathbf{f}_{ic}^\mathsf{T}\right)\mathbf{x} \\
&\qquad - \left(\frac{\mathbf{K}^\mathsf{T}\mathbf{y}}{\sigma^2} + \beta\sum_{c\in\mathcal{C}}\sum_i \mathbf{f}_{ic}\left(\mathbf{f}_{ic}^\mathsf{T}\mathbf{x} - \frac{\rho_i'(\mathbf{f}_{ic}^\mathsf{T}\mathbf{x})}{\beta}\right)\right) \\
&= \mathbf{A}\mathbf{x} - \mathbf{b}(\mathbf{x}),
\end{aligned} \tag{3.111}$$

with

$$\mathbf{A} = \mathbf{K}^\mathsf{T}\mathbf{K}/\sigma^2 + \beta\sum_i \mathbf{F}_i^\mathsf{T}\mathbf{F}_i \tag{3.112}$$

$$\mathbf{b}(\mathbf{x}) = \mathbf{K}^\mathsf{T}\mathbf{y}/\sigma^2 + \beta\sum_i \mathbf{F}_i^\mathsf{T}\,\mathrm{vec}_\mathcal{C}\{\mathbf{f}_{ic}^\mathsf{T}\mathbf{x} - \rho_i'(\mathbf{f}_{ic}^\mathsf{T}\mathbf{x})/\beta\} \tag{3.113}$$

and $\mathbf{F}_i\mathbf{x}$ denoting convolution as usual, where $\mathrm{vec}_\mathcal{C}\{.\}$ is a column vector with entries for $c \in \mathcal{C}$.

As in the multiplicative form, we cannot directly solve $\nabla_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}) = \mathbf{A}\mathbf{x} - \mathbf{b}(\mathbf{x}) = \mathbf{0}$ for $\mathbf{x}$ to find a (local) optimum. However, we can similarly approximate the gradient around the current solution to devise an iterative algorithm. Concretely, we linearly approximate the gradient around $\hat{\mathbf{x}}^{(t-1)}$ as $\nabla_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}) \approx \mathbf{A}\mathbf{x} - \mathbf{b}(\hat{\mathbf{x}}^{(t-1)})$, and iterate as follows:

$$\begin{aligned}\hat{\mathbf{x}}^{(t)} &\leftarrow \mathbf{A}^{-1}\mathbf{b}\big(\hat{\mathbf{x}}^{(t-1)}\big)\\ &= \left(\frac{\mathbf{K}^{\mathsf{T}}\mathbf{K}}{\sigma^2} + \beta\sum_i \mathbf{F}_i^{\mathsf{T}}\mathbf{F}_i\right)^{-1}\\ &\qquad \left(\frac{\mathbf{K}^{\mathsf{T}}\mathbf{y}}{\sigma^2} + \beta\sum_i \mathbf{F}_i^{\mathsf{T}} \operatorname{vec}_C\left\{\mathbf{f}_{ic}^{\mathsf{T}}\hat{\mathbf{x}}^{(t-1)} - \frac{\rho_i'(\mathbf{f}_{ic}^{\mathsf{T}}\hat{\mathbf{x}}^{(t-1)})}{\beta}\right\}\right).\end{aligned} \tag{3.114}$$

This corresponds exactly to the update step in the additive HQ form, as summarized in Fig. 3.6. We can further rewrite this as a gradient descent step

$$\begin{aligned}\hat{\mathbf{x}}^{(t)} &\leftarrow \mathbf{A}^{-1}\mathbf{b}\big(\hat{\mathbf{x}}^{(t-1)}\big)\\ &= \mathbf{A}^{-1}\Big(\mathbf{A}\hat{\mathbf{x}}^{(t-1)} - \mathbf{A}\hat{\mathbf{x}}^{(t-1)} + \mathbf{b}\big(\hat{\mathbf{x}}^{(t-1)}\big)\Big)\\ &= \hat{\mathbf{x}}^{(t-1)} - \mathbf{A}^{-1}\nabla_{\hat{\mathbf{x}}^{(t-1)}} E\big(\hat{\mathbf{x}}^{(t-1)}|\mathbf{y}\big),\end{aligned} \tag{3.115}$$

where $\mathbf{A}$ is a fixed preconditioner that does not change during iterative minimization. Nikolova and Ng [2005] have discussed this and further details under the assumption of convex penalties ρ.

Our previous discussion of the linear gradient approximation in the multiplicative form also applies here, *i.e.* this yields an approximation of the energy function and not a bound as guaranteed with a HQ formulation.

3.4.3 *Summary*

After having introduced the two HQ forms with specific examples, it is evident that using the envelope type is more convenient for MAP estimation, since we can use tools from convex duality to easily obtain update equations for the latent variables. However, as mentioned before, the integral type is necessary if we are interested in probabilistic inference beyond MAP estimation (*cf.* Chapter 4).

The multiplicative form can be used for a larger class of potentials [Palmer et al., 2006] and intuitively seems to provide a better local bound for potential functions that are typically used in practice (*cf.* Fig. 3.4 *vs.* Fig. 3.7). Hence, it might not be surprising that Nikolova and Ng [2005] have shown (theoretically and empirically) that MAP estimation with the multiplicative form convergences in fewer iterations to a good solution as compared with the additive form. However, the additive form typically has much lower computational cost per iteration, and is thus recommended when applicable (Allain et al.

Nikolova and Ng [2005] and Allain et al. [2006] studied convergence under the assumption of convex penalty functions.

[2006] also observe this). Hence, the additive form may overall be much faster than the multiplicative form.

Furthermore, we have shown that half-quadratic MAP estimation is related to several other optimization methods. In particular, especially useful might be the interpretation as an EM algorithm or a specific quasi-Newton (*i. e.*, second-order) optimization method.

3.5 SOLVING EQUATION SYSTEMS

We have seen in the previous sections that half-quadratic inference can be carried out by alternating between two steps: 1) updating independent latent variables z_{ic}, one for each potential $e^{-\rho_i}$ and clique $c \in \mathcal{C}$ of the GM and 2) updating the image $\mathbf{x}$ via a multivariate Gaussian distribution $\mathcal{N}(\mathbf{x}; \mathbf{\Omega}^{-1}\boldsymbol{\eta}, \mathbf{\Omega}^{-1})$, where $\boldsymbol{\eta}$ and the precision matrix $\mathbf{\Omega}$ are easy to compute. The first step is typically simple, since it just consists of solving many one-dimensional optimization problems, which often have a simple analytical solution or just can be precomputed. The second step is more difficult, since the pixels are not independent, but linked via a Gaussian random field. However, its structure is still much simpler than the original random field.

Concretely, the mode of the Gaussian distribution can be obtained by solving the quadratic optimization problem

$$\arg\max_{\mathbf{x}} \mathcal{N}(\mathbf{x}; \mathbf{\Omega}^{-1}\boldsymbol{\eta}, \mathbf{\Omega}^{-1}) = \arg\min_{\mathbf{x}} \frac{1}{2}\mathbf{x}^{\mathsf{T}}\mathbf{\Omega}\mathbf{x} - \mathbf{x}^{\mathsf{T}}\boldsymbol{\eta}, \qquad (3.116)$$

where the solution can be characterized by a system of linear equations:

$$\nabla_{\mathbf{x}}\left(\frac{1}{2}\mathbf{x}^{\mathsf{T}}\mathbf{\Omega}\mathbf{x} - \mathbf{x}^{\mathsf{T}}\boldsymbol{\eta}\right) = 0 \quad \Rightarrow \quad \mathbf{\Omega}\mathbf{x} = \boldsymbol{\eta}. \qquad (3.117)$$

Obviously, the solution $\mathbf{x} = \mathbf{\Omega}^{-1}\boldsymbol{\eta}$ of this equation system is just the mean of the Gaussian distribution, which can be obtained by inverting the precision matrix $\mathbf{\Omega}$. However, matrix inversion is a computationally costly operation, which we would like to avoid. Furthermore, while $\mathbf{\Omega}$ is a sparse matrix with relatively few non-zero elements as determined by the connectivity of the underlying GM, the inverse $\mathbf{\Omega}^{-1}$ is typically a dense matrix with n^2 entries, when $\mathbf{x} \in \mathbb{R}^n$ is an image with n pixels, thus incurring a prohibitive memory cost for large images. Drawing samples from $\mathcal{N}(\mathbf{x}; \mathbf{\Omega}^{-1}\boldsymbol{\eta}, \mathbf{\Omega}^{-1})$ can be accomplished by solving similar equation systems (Section 3.5.3), and may be interpreted as finding the mode of a Gaussian random field with "perturbed" potentials [Papandreou and Yuille, 2010]. Hence, the following discussion will equally apply to finding the mode of a Gaussian in the context of MAP estimation and to drawing samples.

POSITIVE-DEFINITE MATRICES So far, we have implicitly assumed that $\mathbf{\Omega}^{-1}$ is a symmetric *positive-definite* (SPD) matrix, otherwise the

Gaussian density $\mathcal{N}(\mathbf{\Omega}^{-1}\boldsymbol{\eta}, \mathbf{\Omega}^{-1})$ is said to be *degenerate*. A symmetric and real-valued matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ is defined to be positive-definite if $\forall \mathbf{x} \neq \mathbf{0} : \mathbf{x}^\mathsf{T}\mathbf{M}\mathbf{x} > 0$. It can further be shown [*cf.* Boyd and Vandenberghe, 2004, § 3.1.4] that the quadratic function

$$f(\mathbf{x}) = \mathbf{x}^\mathsf{T}\mathbf{M}\mathbf{x} + \mathbf{x}^\mathsf{T}\mathbf{b} + c \tag{3.118}$$

is *strictly* convex $\forall \mathbf{x}, \mathbf{b} \in \mathbb{R}^n, c \in \mathbb{R}$ if $\mathbf{M}$ is SPD. Additionally, every SPD matrix $\mathbf{M}$ is invertible, and its inverse $\mathbf{M}^{-1}$ is also SPD. Hence, if $\mathbf{\Omega}^{-1}$ is SPD then so is $\mathbf{\Omega}$, and the quadratic problem in Eq. (3.116) is strictly convex, which in turn guarantees a unique solution [*cf.* Boyd and Vandenberghe, 2004, § 4.2.3], which can be found by solving the system of linear equations in Eq. (3.117).

To investigate the conditions under which our precision matrices are SPD, let us consider the matrix

$$\mathbf{\Omega} = \mathbf{K}^\mathsf{T}\mathbf{\Lambda}\mathbf{K} + \sum_{i=1}^{N} \mathbf{F}_i^\mathsf{T}\mathbf{Z}_i\mathbf{F} \tag{3.119}$$

where $\mathbf{\Lambda}$ and $\mathbf{Z}_i$ are diagonal matrices with all positive entries (here, typically $\mathbf{\Lambda} = \mathbf{I}/\sigma^2$). This matrix is representative of the precision matrices that we encountered so far and will encounter throughout the remainder of this thesis (with the exception of Chapter 6). In the multiplicative form, $\mathbf{Z}_i = \mathcal{D}_\mathcal{C}\{z_{ic}\}$ is a diagonal matrix with elements $z_{ic} > 0$. In the additive form, $\mathbf{Z}_i = \beta\mathbf{I}$ with $\beta > 0$. If we define

$$\mathbf{W} = \left[\mathbf{F}_1^\mathsf{T}, \ldots, \mathbf{F}_N^\mathsf{T}, \mathbf{K}^\mathsf{T}\right] \in \mathbb{R}^{n \times m} \tag{3.120}$$

$$\mathbf{D} = \mathcal{D}\{\mathbf{Z}_1, \ldots, \mathbf{Z}_N, \mathbf{\Lambda}\} \in \mathbb{R}^{m \times m} \tag{3.121}$$

with $\mathbf{x} \in \mathbb{R}^n$ and $m > n$, then we can express the precision matrix as

$$\mathbf{\Omega} = \mathbf{W}\mathbf{D}\mathbf{W}^\mathsf{T} = \begin{bmatrix} \mathbf{F}_1 \\ \vdots \\ \mathbf{F}_N \\ \mathbf{K} \end{bmatrix}^\mathsf{T} \begin{bmatrix} \mathbf{Z}_1 & & \cdots & \mathbf{0} \\ & \ddots & & \vdots \\ \vdots & & \mathbf{Z}_N & \\ \mathbf{0} & \cdots & & \mathbf{\Lambda} \end{bmatrix} \begin{bmatrix} \mathbf{F}_1 \\ \vdots \\ \mathbf{F}_N \\ \mathbf{K} \end{bmatrix}. \tag{3.122}$$

Note that $\mathbf{D}$ is a large diagonal matrix with only positive elements, and as such an SPD matrix. Furthermore, $\mathbf{W}\mathbf{D}\mathbf{W}^\mathsf{T} \in \mathbb{R}^{n \times n}$ is SPD if $\mathbf{W} \in \mathbb{R}^{n \times m}$ has rank n [*cf.* Golub and van Loan, 1996, § 4.2.1]. Consequently, $\mathbf{W}$ has rank n if only one of the matrices $\mathbf{F}_1, \ldots, \mathbf{F}_N, \mathbf{K}$ is square and of rank n, which for instance is the case in image denoising with $\mathbf{K} = \mathbf{I}$ being an identity matrix. Since $\mathbf{W}$ could have less than rank n and $\mathbf{\Omega}$ would thus not be SPD, in practice we often instead use $\widetilde{\mathbf{W}} = [\mathbf{W}, \mathbf{I}]$ and block diagonal matrix $\widetilde{\mathbf{D}} = \mathcal{D}\{\mathbf{D}, \epsilon\mathbf{I}\}$ with $\epsilon > 0$ being a very small constant, which results in a modified matrix $\widetilde{\mathbf{\Omega}} = \widetilde{\mathbf{W}}\widetilde{\mathbf{D}}\widetilde{\mathbf{W}}^\mathsf{T} = \mathbf{\Omega} + \epsilon\mathbf{I}$ that is guaranteed to be SPD (since $\widetilde{\mathbf{W}}$ has rank n). When employing this regularization, it implies that we use a slightly modified prior $\widetilde{p}(\mathbf{x}) \propto \exp(-\epsilon\|\mathbf{x}\|^2/2) \cdot p(\mathbf{x})$.

Such an approach is actually necessary for drawing samples from the image prior, due to the matrices $\mathbf{F}_i$ being convolution matrices defined by linear *zero-sum* filters $\mathbf{f}_i$, *i. e.* $\mathbf{f}_i^\mathsf{T}\mathbf{1} = 0$. As a result, we locally have $\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)} = \mathbf{f}_i^\mathsf{T}(\mathbf{x}_{(c)} + c)$ for any constant $c \in \mathbb{R}$ and the prior thus globally has the property $p(\mathbf{x}) = p(\mathbf{x} + c)$. After augmenting the prior with latent variables $\mathbf{z}$, the conditional $p(\mathbf{x}|\mathbf{z})$ is actually a degenerate Gaussian distribution since the integral over $\mathbf{x}$ does not exist. This implies that the precision matrix of $p(\mathbf{x}|\mathbf{z})$ is not SPD since the rank of the matrix $[\mathbf{F}_1, \dots, \mathbf{F}_N]$ is less than n. However, using $\widetilde{p}(\mathbf{x})$ instead of $p(\mathbf{x})$ addresses this since it holds that $\widetilde{p}(\mathbf{x}) \neq \widetilde{p}(\mathbf{x} + c)$.

ITERATIVE AND DIRECT SOLVERS Having established that $\mathbf{\Omega}$ is an SPD matrix, we can either solve the equation system $\mathbf{\Omega x} = \boldsymbol{\eta}$ by using an *iterative* or a *direct* method. Direct methods first factorize the equation system matrix $\mathbf{\Omega}$ as the product of simpler matrices, *e. g.* triangular and diagonal ones, which can then be used to quickly and easily solve the equation system. Iterative methods, on the other hand, start with an initial solution provided by the user and then iteratively improve it such that the residual error is reduced in every step.

Especially for smaller problems, direct methods are typically faster and have a predictable runtime that does not depend on the entries of the matrix $\mathbf{\Omega}$. However, they often cannot be applied to large problems (images), where computing and/or storing a matrix factorization is impractical. For these large problems, iterative methods can still be applied and typically do not even need access to the matrix $\mathbf{\Omega}$, but only require as input a function $f(\mathbf{x}) = \mathbf{\Omega x}$ that computes matrix-vector products. Philosophically, one may think of direct methods as solving the equation system of Eq. (3.117) and iterative methods as solving the quadratic optimization problem of Eq. (3.116).

3.5.1 *Matrix factorizations*

3.5.1.1 *Cholesky decomposition*

Given an SPD matrix $\mathbf{\Omega} \in \mathbb{R}^{n \times n}$, the *Cholesky decomposition* produces the factorization

$$\mathbf{\Omega} = \mathbf{L}\mathbf{L}^\mathsf{T} \tag{3.123}$$

where $\mathbf{L} \in \mathbb{R}^{n \times n}$ is a unique and invertible *lower triangular* matrix, *i. e.* all elements above the main diagonal are zero. Consequently, $\mathbf{L}^\mathsf{T}$ is an *upper triangular* matrix. Solving equation systems with lower (upper) triangular matrices can easily be accomplished by forward (backward) substitution. Concretely, we can address the equation system $\mathbf{\Omega x} = \mathbf{L}\mathbf{L}^\mathsf{T}\mathbf{x} = \boldsymbol{\eta}$ by first solving $\mathbf{Lu} = \boldsymbol{\eta}$ via forward substitution to obtain a temporary result $\mathbf{u}$, and then obtain the solution $\mathbf{x}$ by solving $\mathbf{L}^\mathsf{T}\mathbf{x} = \mathbf{u}$ via backward substitution, *i. e.*

$$\mathbf{x} = \boldsymbol{\Omega}^{-1}\boldsymbol{\eta} = \mathbf{L}^{-\top}\mathbf{L}^{-1}\boldsymbol{\eta} = \mathbf{L}^{-\top}\mathbf{u}, \qquad (3.124)$$

where $\mathbf{L}^{-\top} = (\mathbf{L}^{-1})^\top = (\mathbf{L}^\top)^{-1}$. Note that in the additive form, we only need to compute the Cholesky decomposition once and can then use it in all iterations of half-quadratic inference, since the matrix does not depend on the changing latent variables. This is in contrast to the multiplicative form, where we have to compute the decomposition in every iteration.

For general (dense) matrices, a Cholesky decomposition has computational complexity $\mathcal{O}(n^3)$, whereas forward and backward substitution both require $\mathcal{O}(n^2)$ operations. Since n denotes the number of image pixels in our targeted applications, this seems prohibitive. Fortunately, there are variants of the Cholesky decomposition for sparse (banded) matrices [*cf.* Rue and Held, 2005], which produce factorizations with sparse $\mathbf{L}$ and reduced runtime that depends on the number and structure of non-zero matrix elements. The runtime of forward/backward substitution is also decreased for sparse Cholesky factors $\mathbf{L}$.

Hence, we can directly use this approach in the context of MAP estimation to find the mode of the Gaussian distribution. Additionally, a Cholesky decomposition will also be useful for sampling.

3.5.1.2 *Other decompositions*

Disadvantages of the Cholesky decomposition are that the sparse equation system matrix $\boldsymbol{\Omega}$ must be explicitly constructed and that the obtained matrix $\mathbf{L}$ is often less sparse compared to $\boldsymbol{\Omega}$. A Cholesky decomposition is thus often impractical for large problems due to computational and memory demands.

In the previous section, we have already encountered a factorization of the precision matrix in Eq. (3.122) as $\boldsymbol{\Omega} = \mathbf{W}\mathbf{D}\mathbf{W}^\top$ with diagonal matrix $\mathbf{D} \in \mathbb{R}^{m \times m}$ and sparse matrix $\mathbf{W} \in \mathbb{R}^{n \times m}$. Hence, we can define $\mathbf{H} = \mathbf{W}\sqrt{\mathbf{D}} \in \mathbb{R}^{n \times m}$ with $\boldsymbol{\Omega} = \mathbf{H}\mathbf{H}^\top$. However, note that $\mathbf{H}$ neither has favorable structure nor is a square matrix, thus we cannot use the same approach to solve the equation system as we have done with the Cholesky decomposition. However, the factorization $\boldsymbol{\Omega} = \mathbf{W}\mathbf{D}\mathbf{W}^\top$ will be useful in the context of sampling.

BLOCK-CIRCULANT DECOMPOSITION In the additive half-quadratic form only, we can additionally use another very effective decomposition, when $\mathbf{F}_i\mathbf{x} \equiv \mathbf{f}_i \otimes \mathbf{x}$ and $\mathbf{K}\mathbf{x} \equiv \mathbf{k} \otimes \mathbf{x}$, where $\otimes$ denotes two-dimensional convolution with periodic (circular, wrap-around) boundary conditions. If this is the case, then the matrices $\mathbf{K}, \mathbf{F}_i$ have a special structure that is called *block circulant with circulant blocks* (BCCB) [*cf.* Gray, 2006]. A BCCB matrix $\mathbf{B}$ can be decomposed as

$$\mathbf{B} = \mathcal{F}^{-1}\check{\mathbf{B}}\mathcal{F}, \qquad (3.125)$$

where $\check{\mathbf{B}}$ is a diagonal matrix that contains the eigenvalues of $\mathbf{B}$ and $\mathcal{F}$ denotes (the matrix that corresponds to) the two-dimensional unitary discrete Fourier transform (DFT). Note that $\mathcal{F}^{-1} = \mathcal{F}^*$ where $\mathcal{F}^* = \overline{\mathcal{F}}^{\mathsf{T}}$ denotes the conjugate transpose with $\overline{\mathcal{F}}$ being the matrix $\mathcal{F}$ with complex conjugated entries. The matrix $\check{\mathbf{B}} \equiv \mathcal{F}(\mathbf{b})$ is also known as the *optical transfer function* of the *point spread function* (*i. e.*, linear filter) $\mathbf{b}$. We can easily verify that multiplication by $\mathbf{B}$ corresponds to periodic convolution with $\mathbf{b}$ by using Eq. (3.125) and the well-known *convolution theorem*:

$$\mathbf{B}\mathbf{x} = \mathcal{F}^{-1}\check{\mathbf{B}}\mathcal{F}\mathbf{x} \equiv \mathcal{F}^{-1}(\mathcal{F}\mathbf{b} \cdot \mathcal{F}\mathbf{x}) = \mathbf{b} \otimes \mathbf{x}. \tag{3.126}$$

Assuming that all matrices $\mathbf{K}, \mathbf{F}_i$ are BCCB, the precision matrix in the additive form can be factorized as

$$\begin{aligned}
\mathbf{\Omega} &= \mathbf{K}^*\mathbf{K}/\sigma^2 + \beta \sum_{i=1}^{N} \mathbf{F}_i^*\mathbf{F} \\
&= \frac{1}{\sigma^2}(\mathcal{F}^*\check{\mathbf{K}}\mathcal{F})^*(\mathcal{F}^*\check{\mathbf{K}}\mathcal{F}) + \beta \sum_{i=1}^{N} (\mathcal{F}^*\check{\mathbf{F}}_i\mathcal{F})^*(\mathcal{F}^*\check{\mathbf{F}}_i\mathcal{F}) \\
&= \mathcal{F}^*\left(\frac{1}{\sigma^2}|\check{\mathbf{K}}|^2 + \beta \sum_{i=1}^{N} |\check{\mathbf{F}}_i|^2\right)\mathcal{F} \\
&= \mathcal{F}^*\check{\mathbf{D}}\mathcal{F}
\end{aligned} \tag{3.127}$$

and is thus also BCCB, where we have defined the diagonal matrix $\check{\mathbf{D}} = |\check{\mathbf{K}}|^2/\sigma^2 + \beta \sum_{i=1}^{N} |\check{\mathbf{F}}_i|^2$ and we initially replaced the normal transpose operator with the conjugate transpose, which is equivalent for real-valued matrices. The obtained decomposition can also be interpreted as a basis transform, such that the matrix $\mathbf{\Omega}$ is diagonal w.r.t. the new basis. Here, the basis transform is easy to compute via DFTs.

This decomposition is well suited for solving the system of linear equations that arises in the context of MAP estimation as

$$\mathbf{x} = \mathbf{\Omega}^{-1}\boldsymbol{\eta} = (\mathcal{F}^*\check{\mathbf{D}}\mathcal{F})^{-1}\boldsymbol{\eta} = \mathcal{F}^*\check{\mathbf{D}}^{-1}\mathcal{F}\boldsymbol{\eta}, \tag{3.128}$$

which can be done with a complexity of $\mathcal{O}(n \log n)$, since computing the DFTs is the most expensive operation. Furthermore, the matrix $\check{\mathbf{D}}$ can be pre-computed once and then be used throughout half-quadratic inference, since it implicitly only depends on the size of the image $\mathbf{x}$, but not the image itself nor the latent variables $\mathbf{z}$. Only the RHS $\boldsymbol{\eta} \equiv \boldsymbol{\eta}_{\mathbf{z}}$ of the equation system depends on $\mathbf{z}$ and thus varies during half-quadratic inference. In practice, solving the equation system in this way can be orders of magnitude faster than using a Cholesky decomposition or an iterative method, especially for large problems. Furthermore, DFTs are well-suited to parallel execution on graphics processing units (GPUs) for additional speedups.

In case the precision matrix is almost BCCB, but not exactly, which for instance is the case when convolution is not carried out with periodic boundary conditions, one may exploit the above (or similar) factorization as a *preconditioner* for an iterative solver (*cf.* Section 3.5.2; Chan and Ng [1996] discuss the case of (block) *Toeplitz* matrices and applications to image restoration). Alternatively, one can split the non-BCCB precision matrix

$$\mathbf{\Omega} = \mathcal{F}^* \check{\mathbf{D}} \mathcal{F} + \mathbf{U}\mathbf{U}^\mathsf{T} \tag{3.129}$$

into a BCCB matrix as above plus a low-rank matrix $\mathbf{U}\mathbf{U}^\mathsf{T}$ and then use the *Woodbury matrix identity* [*cf.* Hager, 1989] to efficiently solve the resulting equation system [*cf.* Jain, 1978]. Another possibility may be to alter the HQ construction by applying an explicit correction to make the precision matrix BCCB; such an approach has been taken by Husse et al. [2004].

3.5.2 *Iterative solvers and preconditioners*

Although general-purpose matrix factorizations, such as the Cholesky decomposition, are very effective at solving equation systems to high accuracy, they are unfortunately often not applicable to large problems due to prohibitive computation and memory requirements. More efficient matrix-specific factorizations are not always applicable (here especially in the multiplicative HQ form). Hence, in contrast to directly solving the equation system Eq. (3.117) by factorizing the system matrix, we can alternatively minimize the quadratic function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\mathsf{T}\mathbf{\Omega}\mathbf{x} - \mathbf{x}^\mathsf{T}\boldsymbol{\eta}, \tag{3.130}$$

of Eq. (3.116). Since f is differentiable and strictly convex, we can use gradient-based descent algorithms that via a sequence of approximate solutions $\mathbf{x}_0, \ldots, \mathbf{x}_k, \ldots, \mathbf{x}_K$ are guaranteed to iteratively converge to the global minimum $\mathbf{x}_K = \arg\min_\mathbf{x} f(\mathbf{x})$ after some K steps. To that end, we improve the solution in each iteration as

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{d}_k \tag{3.131}$$

with descent direction $\mathbf{d}_k$ and step length α_k, such that $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$ for all non-optimal $\mathbf{x}_k$. When choosing the function's gradient

$$\mathbf{g}_k = \nabla_{\mathbf{x}_k} f(\mathbf{x}_k) = \mathbf{\Omega}\mathbf{x}_k - \boldsymbol{\eta} \tag{3.132}$$

as the descent direction with $\mathbf{d}_k = \mathbf{g}_k$, we obtain the well-known method of (steepest) *gradient descent*. The associated step size can be analytically determined via an exact *line search* as

$$\alpha_k = \arg\min_\alpha f(\mathbf{x}_k - \alpha\mathbf{d}_k) = \frac{\mathbf{g}_k^\mathsf{T}\mathbf{d}_k}{\mathbf{d}_k^\mathsf{T}\mathbf{\Omega}\mathbf{d}_k}. \tag{3.133}$$

Unfortunately, gradient descent can have a very slow rate of convergence [*cf.* Nocedal and Wright, 1999, § 3.3].

It turns out that a better choice is

$$\mathbf{d}_k = \mathbf{g}_k - \frac{\mathbf{g}_k^\mathsf{T} \boldsymbol{\Omega} \mathbf{d}_{k-1}}{\mathbf{d}_{k-1}^\mathsf{T} \boldsymbol{\Omega} \mathbf{d}_{k-1}} \mathbf{d}_{k-1}, \tag{3.134}$$

which is obtained as a linear combination of the gradient $\mathbf{g}_k$ and only the previous descent direction $\mathbf{d}_{k-1}$; the step size α_k is again determined as in Eq. (3.133). It can be shown [*e. g.*, Nocedal and Wright, 1999, § 5.1] that this sequence of directions $\mathbf{d}_1, \ldots, \mathbf{d}_k$ is *conjugate* w.r.t. the matrix $\boldsymbol{\Omega}$, *i. e.* $\forall i \neq j : \mathbf{d}_i^\mathsf{T} \boldsymbol{\Omega} \mathbf{d}_j = 0$, which implies that all descent directions are linearly independent. This in turn means that there can be at most n descent directions, since the vectors $\mathbf{d}_k \in \mathbb{R}^n$ define a basis that spans $\mathbb{R}^n$. Hence, using a descent algorithm with these directions will converge to the solution in at most $K = n$ steps. This algorithm was proposed by Hestenes and Stiefel [1952] and is known as the (linear) *conjugate gradient* (CG) method.

Before we analyze the convergence properties of CG, note that $\boldsymbol{\Omega}$ is only used to compute matrix-vector products. Hence, there is actually no need to explicitly construct the matrix since multiplication can be carried out efficiently using convolutions and pixel-wise operations (*cf.* Eq. 3.119).

Although CG has guaranteed convergence in at most n iterations, convergence is typically much faster in practice, which is fortunate since n is large in our case. To get a better estimate of the rate of convergence, it is useful to define the *condition number* of an SPD matrix $\mathbf{A}$ as

$$\kappa(\mathbf{A}) = \|\mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2 = \frac{e_{\max}(\mathbf{A})}{e_{\min}(\mathbf{A})}, \tag{3.135}$$

where $e_{\max}(\mathbf{A})$ and $e_{\min}(\mathbf{A})$ denote the largest and smallest eigenvalues of $\mathbf{A}$. With this, we can upper bound the error of the current solution $\mathbf{x}_k$ at step k as

$$\|\mathbf{x}_k - \hat{\mathbf{x}}\|_{\boldsymbol{\Omega}} \leq \left(\frac{\sqrt{\kappa(\boldsymbol{\Omega})} - 1}{\sqrt{\kappa(\boldsymbol{\Omega})} + 1} \right)^{2k} \|\mathbf{x}_0 - \hat{\mathbf{x}}\|_{\boldsymbol{\Omega}} \tag{3.136}$$

where $\|\mathbf{v}\|_\mathbf{A} = \sqrt{\mathbf{v}^\mathsf{T} \mathbf{A} \mathbf{v}}$ and $\hat{\mathbf{x}} = \arg\min_\mathbf{x} f(\mathbf{x})$ denotes the true solution [*cf.* Nocedal and Wright, 1999, § 5.1]. Equation (3.136) nicely illustrates which aspects of the problem will have an effect on the number CG iterations: *(1)* Initialization is important, *i. e.* when $\mathbf{x}_0$ is already close to the solution $\hat{\mathbf{x}}$, the RHS of Eq. (3.136) will be smaller. In our case, we may choose the solution from the previous half-quadratic iteration to initialize CG. *(2)* Although the algorithm can stop after n steps or when the gradient $\mathbf{g}_k = \mathbf{0}$, in practice we choose a threshold τ and stop when $\|\mathbf{g}_k\| < \tau$. Hence, choosing τ relatively large will accept a solution with larger left-hand side (LHS) of Eq. (3.136) and thus reduces the number of iterations. This may be a viable approach for MAP estimation, where each HQ step does coordinate ascent on the augmented posterior, hence we do not necessarily have to find the global optimum in each ascent step. In the context of Gibbs sampling, however, even small approximation errors might accumulate over time [*cf.* Gilavert et al., 2015]. *(3)* Finally, we can expect fast convergence when the condition number $\kappa(\mathbf{\Omega})$ is small, thus reducing the RHS of Eq. (3.136).

In general, matrices with a small condition number are called *well-conditioned*, whereas *ill-conditioned* matrices have high condition number. Even a small change of $\boldsymbol{\eta}$ can substantially alter the solution to an equation system $\mathbf{\Omega x} = \boldsymbol{\eta}$ with ill-conditioned $\mathbf{\Omega}$. Unfortunately, ill-conditioned matrices frequently occur in practice, hence transforming the problem to an equation system with a well-conditioned matrix would be advantageous. This is exactly what *preconditioners* aim to do, which we will review below.

3.5.2.1 *Preconditioning*

Assuming that $\mathbf{\Omega}$ is not well-conditioned, we want to solve a related, well-conditioned, problem from which we can easily recover the solution to the original problem of Eq. (3.130). To that end, we define a dependent variable

$$\mathbf{u} = \mathbf{Ax} \tag{3.137}$$

where $\mathbf{A}$ is an invertible matrix so that $\mathbf{x} = \mathbf{A}^{-1}\mathbf{u}$ exists, *i.e.* we can recover $\mathbf{x}$ from $\mathbf{u}$ by solving the equation system $\mathbf{Ax} = \mathbf{u}$. Now we can express our quadratic objective function in terms of $\mathbf{u}$ as

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{u}^\top(\mathbf{A}^{-\top}\mathbf{\Omega}\mathbf{A}^{-1})\mathbf{u} - \mathbf{u}^\top(\mathbf{A}^{-\top}\boldsymbol{\eta}), \tag{3.138}$$

which is minimized by the solution to the equation system

$$(\mathbf{A}^{-\top}\mathbf{\Omega}\mathbf{A}^{-1})\mathbf{u} = \mathbf{A}^{-\top}\boldsymbol{\eta}. \tag{3.139}$$

Hence, we can solve the transformed problem of Eq. (3.138) with the CG algorithm and then recover $\mathbf{x} = \mathbf{A}^{-1}\mathbf{u}$. For this to be a viable strategy, we need to choose $\mathbf{A}$ such that the matrix $\mathbf{A}^{-\top}\mathbf{\Omega}\mathbf{A}^{-1}$ is better conditioned than $\mathbf{\Omega}$, and it must be easy to solve equation systems with $\mathbf{A}$. Thus, we will overall need fewer CG iterations to convergence, but each iteration will be more expensive. The matrix $\mathbf{A}$ is called a preconditioner and is often also used via $\mathbf{M} = \mathbf{A}^\top\mathbf{A}$:

$$\Leftrightarrow \quad \mathbf{M}^{-1}\mathbf{\Omega x} = \mathbf{M}^{-1}\boldsymbol{\eta} \tag{3.140}$$

$$\Leftrightarrow \quad (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{\Omega}(\mathbf{A}^{-1}\mathbf{u}) = (\mathbf{A}^\top\mathbf{A})^{-1}\boldsymbol{\eta} \tag{3.141}$$

$$\Leftrightarrow \quad (\mathbf{A}^{-\top}\mathbf{\Omega}\mathbf{A}^{-1})\mathbf{u} = \mathbf{A}^{-\top}\boldsymbol{\eta}. \tag{3.142}$$

In either case, note that $\mathbf{M}$ or $\mathbf{A}$ do not need to be explicitly represented as matrices, since they are only needed to solve equation systems. When using such a preconditioner with CG, the resulting algorithm is called the *preconditioned conjugate gradient* (PCG) method [*cf.* Nocedal and Wright, 1999, § 5.1].

Ideally, $\mathbf{M}$ (or $\mathbf{A}$) is a "nice" matrix, *i. e.* we can easily use it to solve an equation system, but $\mathbf{M} \approx \boldsymbol{\Omega}$ is similar to the actual equation system matrix. Of course, a Cholesky decomposition would make an ideal preconditioner with its lower triangular matrix $\mathbf{L}$ and thus $\boldsymbol{\Omega} = \mathbf{M} = \mathbf{L}\mathbf{L}^{\mathsf{T}}$. In this case, preconditioned CG would converge after 1 iteration and is thus not really necessary. However, the reason we are considering iterative methods is that matrix factorizations like Cholesky are too expensive for large problems. We can generally obtain a preconditioner for SPD matrices by doing an *incomplete* Cholesky decomposition [*cf.* Saad, 2003], where $\tilde{\mathbf{L}} \approx \mathbf{L}$ is a sparse approximation of $\mathbf{L}$, which can be obtained with reduced computation and storage requirements.

However, the best preconditioners are typically tailored to a specific problem (matrix). For instance, an interesting line of research has been pursued by Krishnan *et al.* [Krishnan and Szeliski, 2011; Krishnan et al., 2013], who devise preconditioner for applications in graphics and computer vision that can be applied to inhomogeneous matrices, as they arise in the multiplicative HQ form, which in our experience can suffer from badly conditioned precision matrices. Unfortunately, their approach can thus far only be applied to image priors that impose smoothness by penalizing differences of pixels in a (small) neighborhood. This includes the popular class of pairwise MRFs but does not generalize to using arbitrary linear filters $\mathbf{f}_i$ of extended size that give rise to high-order FoE priors as considered here.

3.5.3 *Sampling*

In the context of Gibbs sampling (Alg. 3.4) or simulated annealing for MAP estimation (Alg. 3.2), we need to draw a sample from the multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{\Omega}^{-1}\boldsymbol{\eta}, \boldsymbol{\Omega}^{-1})$ with $\boldsymbol{\Omega}$ as defined in Eq. (3.119). This can be accomplished in one of two ways: *1)* As for MAP estimation, we first compute the mean (mode) $\boldsymbol{\mu} = \boldsymbol{\Omega}^{-1}\boldsymbol{\eta}$, then draw a sample from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Omega}^{-1})$, or *2)* we directly draw a sample from $\mathcal{N}(\boldsymbol{\Omega}^{-1}\boldsymbol{\eta}, \boldsymbol{\Omega}^{-1})$ without computing the mean. The first strategy is beneficial when using a more efficient *Rao-Blackwellized* estimator (*cf.* Chapter 4). However, computing the mean in addition to drawing a sample has additional computation cost, hence the second strategy is generally preferable if we are not interested in the Gaussian mean.

Since we can easily sample from a Gaussian distribution with zero mean and diagonal covariance matrix $\mathbf{C}$, we can make use of a well-known property of Gaussian distributions to yield a sample from a different Gaussian by transforming the random vector:

$$\mathbf{r} \sim \mathcal{N}(\mathbf{0}, \mathbf{C}) \quad \Rightarrow \quad (\mathbf{A}\mathbf{r} + \boldsymbol{\mu}) \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{A}\mathbf{C}\mathbf{A}^{\mathsf{T}}). \tag{3.143}$$

Hence, we need $\boldsymbol{\mu} = \boldsymbol{\Omega}^{-1}\boldsymbol{\eta}$, which can be obtained in the same way as explained in the previous section in the context of MAP estimation.

Furthermore, we need to find matrices $\mathbf{A}$ and $\mathbf{C}$, such that $\mathbf{\Omega}^{-1} = \mathbf{A}\mathbf{C}\mathbf{A}^{\mathsf{T}}$. Given such matrices, we can obtain a sample

$$\mathbf{x} = \mathbf{u} + \boldsymbol{\mu} \sim \mathcal{N}(\mathbf{\Omega}^{-1}\boldsymbol{\eta}, \mathbf{\Omega}^{-1}) \tag{3.144}$$

with $\mathbf{r} \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$ and the temporary result $\mathbf{u} = \mathbf{A}\mathbf{r}$. This approach is similarly known as sampling via optimization with *perturbation* [Papandreou and Yuille, 2010; Orieux et al., 2012].

CHOLESKY DECOMPOSITION Given the Cholesky decomposition $\mathbf{\Omega} = \mathbf{L}\mathbf{L}^{\mathsf{T}}$, we can use $\mathbf{A} = \mathbf{L}^{-\mathsf{T}}$ and $\mathbf{C} = \mathbf{I}$ to obtain

$$\mathbf{A}\mathbf{C}\mathbf{A}^{\mathsf{T}} = \mathbf{L}^{-\mathsf{T}}\mathbf{L}^{-1} = (\mathbf{L}\mathbf{L}^{\mathsf{T}})^{-1} = \mathbf{\Omega}^{-1}. \tag{3.145}$$

The temporary vector $\mathbf{u}$ can be obtained by solving the system $\mathbf{L}^{\mathsf{T}}\mathbf{u} = \mathbf{r}$, which can then be added to $\boldsymbol{\mu}$ to obtain the sample from the distribution. Note that we had to solve three triangular equation systems to obtain the sample, assuming that we also used the Cholesky decomposition to compute $\boldsymbol{\mu} = \mathbf{L}^{-\mathsf{T}}\mathbf{L}^{-1}\boldsymbol{\eta}$. This has already been proposed by Rue [2001]. However, if we do not need to compute $\boldsymbol{\mu}$ in a separate step, we can obtain a sample

$$\mathbf{x} = \mathbf{u} + \boldsymbol{\mu} = (\mathbf{L}^{-\mathsf{T}}\mathbf{r}) + (\mathbf{L}^{-\mathsf{T}}\mathbf{L}^{-1}\boldsymbol{\eta}) = \mathbf{L}^{-\mathsf{T}}(\mathbf{r} + \mathbf{L}^{-1}\boldsymbol{\eta}) \tag{3.146}$$

by solving only two equation systems, namely $\mathbf{L}\mathbf{v} = \boldsymbol{\eta}$ for temporary vector $\mathbf{v}$ and then $\mathbf{L}^{\mathsf{T}}\mathbf{x} = \mathbf{r} + \mathbf{v}$ for $\mathbf{x}$.

LARGE PROBLEMS We mentioned before that a (Cholesky) decomposition is deemed intractable for large problems, hence we need to resort to iterative methods (*e.g.*, preconditioned CG) to solve the equation system directly with the matrix $\mathbf{\Omega}$. However, we need a decomposition $\mathbf{A}\mathbf{C}\mathbf{A}^{\mathsf{T}} = \mathbf{\Omega}^{-1}$ for sampling, which may seem unattainable. Fortunately, we can make use of the decomposition $\mathbf{\Omega} = \mathbf{W}\mathbf{D}\mathbf{W}^{\mathsf{T}}$ from Eq. (3.122), because $\mathbf{A}$ does not have to be an invertible matrix here. Concretely, we can use $\mathbf{A} = \mathbf{\Omega}^{-1}\mathbf{W}$ and $\mathbf{C} = \mathbf{D}$ to obtain

$$\mathbf{A}\mathbf{C}\mathbf{A}^{\mathsf{T}} = (\mathbf{\Omega}^{-1}\mathbf{W})\mathbf{D}(\mathbf{\Omega}^{-1}\mathbf{W})^{\mathsf{T}} = \mathbf{\Omega}^{-1}\mathbf{W}\mathbf{D}\mathbf{W}^{\mathsf{T}}\mathbf{\Omega}^{-1} = \mathbf{\Omega}^{-1}. \tag{3.147}$$

This has already been suggested by Levi [2009] and was used by Schmidt et al. [2010]. Similar to above, we can first solve $\mathbf{\Omega}\boldsymbol{\mu} = \boldsymbol{\eta}$ for $\boldsymbol{\mu}$ and then $\mathbf{\Omega}\mathbf{u} = \mathbf{W}\mathbf{r}$ for $\mathbf{u}$ to obtain $\mathbf{x} = \mathbf{u} + \boldsymbol{\mu}$. Alternatively, we directly solve $\mathbf{\Omega}\mathbf{x} = \mathbf{W}\mathbf{r} + \boldsymbol{\eta}$ for $\mathbf{x}$ if we do not need $\boldsymbol{\mu}$, since

$$\mathbf{x} = \mathbf{u} + \boldsymbol{\mu} = (\mathbf{\Omega}^{-1}\mathbf{W}\mathbf{r}) + (\mathbf{\Omega}^{-1}\boldsymbol{\eta}) = \mathbf{\Omega}^{-1}(\mathbf{W}\mathbf{r} + \boldsymbol{\eta}). \tag{3.148}$$

In contrast to direct methods, iterative equation solvers require the user to choose a threshold for the residual error, which greatly influences the number of necessary iterations to achieve the desired solution accuracy. Accepting a relatively high residual error may not be an issue in the context of MAP estimation, but it can be problematic for sampling due to the propagation of errors in the context of MCMC methods, such as the Gibbs sampler in Alg. 3.4. Hence, solving the equation systems with low accuracy may lead to samples that are not representative of the desired target distribution. On the other hand, allowing solutions with low accuracy can drastically reduce the number of required iterations, which amounts to big computational savings for large problems. Gilavert et al. [2015] address this issue by proposing to solve the equation systems with lower accuracy, but to correct for this with a subsequent acceptance/rejection step; the solution accuracy can be tuned to obtain a desired acceptance probability or to minimize the computational cost.

4

BAYESIAN IMAGE RESTORATION WITH UNOBSERVED VARIABLES

CONTENTS

4.1 Estimating Unknown Parameters 91
4.2 Image Restoration 92
  4.2.1 Half-quadratic MRF prior 93
4.3 Posterior Prediction 94
4.4 Bayesian Restoration using Sampling 96
4.5 Noise Estimation 98
4.6 Parametric Blur Estimation 99
4.7 Experiments 101
4.8 Summary 105

IMAGE restoration problems are often addressed by first modeling the corruption process, *i. e.* expressing mathematically how an observed corrupted image $\mathbf{y}$ is related to an unobserved original image $\mathbf{x}$ that should be recovered. In a probabilistic generative setting, this relationship is formalized with a likelihood distribution $p(\mathbf{y}|\mathbf{x})$ that specifies the probability density of observing $\mathbf{y}$ under the assumption that $\mathbf{x}$ is the original image (*cf.* Chapter 1). Although likelihood models are mostly determined by the image formation process (assumed to be known), they often depend on a few instance-specific parameters $\boldsymbol{\beta}$ that can vary for each observed image $\mathbf{y}$, but are typically assumed to be either known or obtained by some preceding estimation step. In this chapter, we treat those parameters as random variables and make the dependence explicit by denoting the likelihood as $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\beta}) \equiv p(\mathbf{y}|\mathbf{x})$. Moreover, although the quality of the restored images often crucially hinges on an appropriate choice for $\boldsymbol{\beta}$ (*cf.* Fig. 4.1), this topic is rarely discussed. We will address this issue in the context of image deblurring and image denoising, where we obtain the restored image by marginalizing over the unknown parameters $\boldsymbol{\beta}$. Moreover, we can additionally compute an estimate of $\boldsymbol{\beta}$ by marginalizing over the restored image.

As mentioned above, most restoration approaches assume that the instance-specific parameters $\boldsymbol{\beta}$ of the likelihood are known or estimated beforehand. However, this can be challenging, such as estimating the Gaussian noise strength from an observed blurred image. Furthermore, generative restoration approaches have for the most part relied on MAP estimation for inference [Roth and Black, 2009; Krishnan

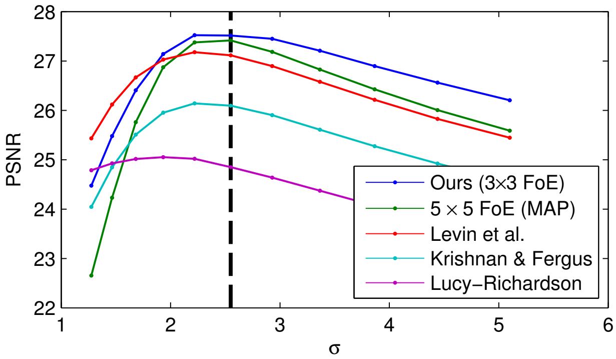


Figure 4.1: **Deblurring results for various assumed Gaussian noise levels.** Average results (based on 8 images) for several deblurring methods under the assumption of various noise levels (correct value $\sigma = 2.55$, depicted as dashed vertical line). The results show that all methods are sensitive to an incorrect value of σ.

and Fergus, 2009; Levin et al., 2007]. As discussed in Chapter 2, this is problematic since the MAP estimate does not reflect a suitable loss for image restoration problems. To alleviate this issue, a regularization weight is typically employed to calibrate the influence between prior and likelihood (*cf.* Section 4.3). Not only does this regularization weight have to be tuned to yield good restoration results, but it can also depend on the instance-specific likelihood parameters (*e.g.*, the Gaussian noise level). In this case, a separate regularization weight has to be specified for each value of the instance-specific likelihood parameters, which can necessarily not be done exhaustively.

In this chapter, we propose an image restoration method that extends the conventional Bayesian approach with an *integrated estimation* of instance-specific likelihood parameters. In particular, we focus on image denoising and non-blind deblurring with integrated noise estimation, where we treat the noise level as an unobserved random variable that can be integrated out using a sampling-based algorithm. As a consequence of combining noise estimation and image restoration, manual noise selection or a separate pre-processing step are no longer needed. However, our approach is not limited to estimating the noise level. We demonstrate this by extending our approach to the case of *parametric* blur, where we assume the kind of blur to be known (*e.g.*, Gaussian blur), but which still depends on a few instance-specific parameters.

Concretely, we employ the learned pairwise and high-order MRF priors of Schmidt et al. [2010], which are based on the Field of Experts (FoE) model [Roth and Black, 2009]. We also adopt and extend inference based on half-quadratic Gibbs sampling (Section 3.3.2.2), which had previously been used for image denoising [Schmidt et al., 2010] and deblurring [Schmidt et al., 2011]. Since we use suitable image priors, we follow Schmidt et al. [2010, 2011] and employ an (approximate) Bayes estimator for the MMSE to obtain the restored images (*cf*. Section 4.3). As a result, we do not need to employ a regularization parameter [*cf*. Schmidt et al., 2010].

We quantitatively compare our blind denoising and non-blind deblurring results with integrated noise estimation to the case where the noise level is known [Schmidt et al., 2010, 2011]; we find in both cases that almost the same performance can be obtained without relying on a known noise level. In addition, we evaluate the noise estimation component itself since we can also obtain a noise estimate with our approach. Finally, we show qualitative blind deblurring results under the assumption of parametric blur for two examples: Gaussian blur and linear (camera) motion. In both cases, we estimate the restored image, the noise level, and the blur parameters.

4.1 ESTIMATING UNKNOWN PARAMETERS

Image priors are typically used in the context of MAP estimation, which is problematic in the generative case as we discussed in Chapter 2. Hence, we use the image priors of Schmidt et al. [2010] and adopt their inference approach based on sampling with a half-quadratic block Gibbs sampler, which has previously been used for denoising [Schmidt et al., 2010] and non-blind deblurring [Schmidt et al., 2011]. This allows us to go beyond MAP estimation and use more suitable estimates, such as the MMSE. Importantly, the Gibbs sampler can rather naturally be extended to also estimate several unknown parameters (such as the noise level).

In contrast, MAP approaches with inference via energy minimization require the noise level and/or regularization parameter to be known or estimated separately. This has been addressed using variational Bayesian techniques that approximate the posterior by a simpler, analytically tractable density. One can thus compute marginal expectations of the hidden variables, including the noise level, under the approximative distribution. Miskin and MacKay [2000] propose a variational Bayesian framework for blind deconvolution with integrated noise estimation, but assume that pixels are i.i.d., which leads to sub-par deblurring results. Fergus et al. [2006] incorporate a similar automatic noise estimate into kernel estimation, but do so for noise on the image gradients instead of image noise. In contrast, our non-blind deblurring algorithm based on sampling allows to formulate and estimate sensor noise in the spatial domain.

The issue of estimating regularization parameters extends well beyond deblurring. In stereo, Zhang and Seitz [2007] address this by performing joint MAP estimation of the disparity and the MRF parameters. In optical flow, Krajsek and Mester [2006] marginalize over the flow field based on a Laplace approximation in order to obtain a maximum marginal likelihood estimate for the model parameters.

Since many image restoration approaches involve noise-dependent tuning parameters, some effort has gone into automatic noise estimation. For a single color image, Liu et al. [2006] infer the noise level in RGB channels using a piecewise smooth image model. For gray-level images, Zoran and Weiss [2009] estimate the noise standard deviation by modeling a link between kurtosis values and image noise. The wavelet-based approach of De Stefano et al. [2004] follows similar ideas. The widely used MAD framework [Donoho and Johnstone, 1994; Zlokolica et al., 2006] infers a noise estimate from the wavelet coefficients of the highest-frequency sub-band. We note that most noise estimation procedures do not explicitly consider the special case of noise inference on blurred (or otherwise corrupted) images. One exception is [Zoran and Weiss, 2009], which at least report experimental results for this case. The advantage of our integrated noise estimation approach is that it is directly applicable to the given application (here, deblurring and denoising).

4.2 IMAGE RESTORATION

As in Chapter 3, we consider image restoration problems that can be modeled with a likelihood of the form

$$p(\mathbf{y}|\mathbf{x}, \mathbf{K}, \sigma) = \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2 \mathbf{I}), \tag{4.1}$$

but here focus on image denoising and deblurring. As usual, $\mathbf{y} \in \mathbb{R}^n$ is the observed, corrupted image and $\mathbf{x} \in \mathbb{R}^m$ denotes the restored image that we want to recover. For image denoising, $\mathbf{K} = \mathbf{I}$ is an identity matrix, *i. e.* the clean image is only contaminated with additive white Gaussian noise of variance σ^2. The deblurring problem is more difficult, where $\mathbf{K} \in \mathbb{R}^{n \times m}$ is a blur matrix. Although our approach can be applied to arbitrary (including non-uniform) blurs $\mathbf{K}$, note that we conduct our experiments (Section 4.7) in the context of uniform blur, where $\mathbf{K}\mathbf{x} \equiv \mathbf{k} \otimes \mathbf{x}$ corresponds to a convolution of $\mathbf{x}$ with the blur kernel $\mathbf{k}$.

In order to compute the restored image, the typical Bayesian approach is to obtain the posterior as

$$p(\mathbf{x}|\mathbf{y}, \mathbf{K}, \sigma) \propto p(\mathbf{y}|\mathbf{x}, \mathbf{K}, \sigma) \cdot p(\mathbf{x}), \tag{4.2}$$

where $p(\mathbf{x})$ denotes a natural image prior that we discuss below in more detail. While Equation (4.2) is the foundation for image denoising and non-blind deblurring in this chapter, we will drop the reliance on σ and extend the posterior to $p(\mathbf{x}, \sigma|\mathbf{y}, \mathbf{K})$, which we use to estimate both the restored image and the Gaussian noise level. In addition, we will go one step further and use $p(\mathbf{x}, \sigma, \boldsymbol{\omega}|\mathbf{y})$ for inference in case of parametric blur, *i. e.* when $\mathbf{K}$ is fully specified by a few parameters $\boldsymbol{\omega}$.

4.2.1 *Half-quadratic MRF prior*

As discussed in Chapter 1, regularization is crucial to address ill-posed image restoration problems. To that end, sparse image priors are often used (*cf.* Section 2.2). Instead of commonly-used hand-defined image priors [*e.g.*, Krishnan and Fergus, 2009; Levin et al., 2007], we here rely on learned (high-order) FoE priors [Roth and Black, 2009]. In particular, we make use of the learned priors from Schmidt et al. [2010], which have already been used for denoising [Schmidt et al., 2010] and non-blind deblurring [Schmidt et al., 2011] under the assumption that the Gaussian noise level is known.

Similar to Section 2.2.2, the FoE prior is defined as

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \prod_{i=1}^{N} \exp\left(-\rho\left(\mathbf{f}_i^{\mathsf{T}} \mathbf{x}_{(c)}; \boldsymbol{\alpha}_i\right)\right), \tag{4.3}$$

but we assume here that each potential/expert function associated to filter $\mathbf{f}_i$ belongs to the same family specified by learned parameters $\boldsymbol{\alpha}_i$. Concretely, we use Gaussian scale mixtures (GSMs) [Wainwright and Simoncelli, 2000]

$$\exp\left(-\rho(\mathbf{f}_i^{\mathsf{T}} \mathbf{x}_{(c)}; \boldsymbol{\alpha}_i)\right) = \sum_{j=1}^{J} \alpha_{ij} \mathcal{N}(\mathbf{f}_i^{\mathsf{T}} \mathbf{x}_{(c)}; 0, \eta_{ij}^2) \tag{4.4}$$

with a fixed number of mixture components, as have been used by Schmidt et al. [2010]; they showed that their learned image priors with GSM experts exhibit good generative properties. Importantly, GSM experts trivially admit the construction of a half-quadratic augmented image prior via the multiplicative form (Section 3.4.1) of the integral type (Section 3.3.2). In particular, the augmented FoE prior can be obtained as

$$p(\mathbf{x}, \mathbf{z}) \propto \prod_{c \in \mathcal{C}} \prod_{i=1}^{N} \alpha_{iz_{ic}} \mathcal{N}(\mathbf{f}_i^{\mathsf{T}} \mathbf{x}_{(c)}; 0, \eta_{iz_{ic}}^2) \tag{4.5}$$

with discrete latent variables $\mathbf{z}$ (one for each expert and clique) that act as indicator variables for the Gaussian mixture components. It is easy to verify that the FoE prior from Eq. (4.3) is retained by marginalizing over $\mathbf{z}$ in Eq. (4.5).

Recall that the benefit of augmented prior $p(\mathbf{x}, \mathbf{z})$ is that the conditional distributions are comparatively easy to work with: $p(\mathbf{x}|\mathbf{z})$ is a multivariate Gaussian and $p(\mathbf{z}|\mathbf{x})$ is a product of univariate discrete distribution [*cf.* Schmidt et al., 2010]. This benefit is also retained for the augmented posterior $p(\mathbf{x}, \mathbf{z}|\mathbf{y}, \mathbf{K}, \sigma) \propto p(\mathbf{y}|\mathbf{x}, \mathbf{K}, \sigma) p(\mathbf{x}, \mathbf{z})$. As a result, we can make use of efficient Gibbs sampling-based inference (Section 3.3.2.2), which we later also extend for our integrated noise (and blur) estimation approach.

Before we discuss our concrete inference approach based on the augmented posterior in Section 4.4, we take a step back and discuss which estimation approach to take. In general, after forming the posterior distribution $p(\mathbf{x}|\mathbf{y}, \mathbf{K}, \sigma)$, we typically want to predict a single restored image given the observed image $\mathbf{y}$ and blur matrix $\mathbf{K}$; for now, we also assume that the noise level σ is given. Since the following discussion is not limited to image deblurring, we drop the dependence on $\mathbf{K}$ and σ and simply use $p(\mathbf{x}|\mathbf{y})$ to denote the posterior distribution, which also declutters the notation and makes the exposition more readable.

If we assume that the posterior distribution is *accurate*, recall from Section 2.3.1 that the optimal prediction

$$\hat{\mathbf{x}} = \arg\min_{\tilde{\mathbf{x}}} \mathbb{E}\left[\Delta(\tilde{\mathbf{x}}, \mathbf{x})|\mathbf{y}\right] = \arg\min_{\tilde{\mathbf{x}}} \int \Delta(\tilde{\mathbf{x}}, \mathbf{x}) p(\mathbf{x}|\mathbf{y})\, d\mathbf{x} \tag{4.6}$$

is obtained by minimizing the expected loss, where $\Delta(\tilde{\mathbf{x}}, \mathbf{x})$ is a *suitable* loss function for the given application.

MAP ESTIMATION However, most approaches based on MRF image priors [*e.g.*, Levin et al., 2007; Krishnan and Fergus, 2009] predict the restored image via MAP estimation as $\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$, which implicitly assumes the 0-1 loss function $\Delta(\tilde{\mathbf{x}}, \mathbf{x}) = \mathbb{I}[\tilde{\mathbf{x}} \neq \mathbf{x}]$. Furthermore, most (manually defined) image priors do not possess good generative properties [Schmidt et al., 2010]. Hence, they do not accurately model the prior, which leads to a misspecified posterior distribution [*cf.* Pletscher et al., 2011]. To compensate for both of these issues, a (noise-dependent) regularization weight λ is typically used, which leads to a posterior that is modified in the following way:

$$p_\lambda(\mathbf{x}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}) \cdot p(\mathbf{x})^\lambda. \tag{4.7}$$

Based on a set of input-output examples, λ is then chosen to improve the MAP estimate of $p_\lambda(\mathbf{x}|\mathbf{y})$ w.r.t. the image quality measure of interest [*e.g.*, Roth, 2007], typically the PSNR. Note that this essentially corresponds to (discriminative) training of a loss-specific regression function via a single parameter λ (*cf.* Section 2.4.1). Hence, choosing the parameter λ can be interpreted as (admittedly weak) discriminative tuning of an apparently generative approach.

MMSE ESTIMATION In contrast to the 0-1 loss, recall from Section 2.6.2 that a more appropriate loss function for image restoration is the *mean squared error* (MSE)

$$\Delta(\tilde{\mathbf{x}}, \mathbf{x}) = \frac{1}{m}\|\tilde{\mathbf{x}} - \mathbf{x}\|^2 = \frac{1}{m}(\tilde{\mathbf{x}} - \mathbf{x})^\mathsf{T}(\tilde{\mathbf{x}} - \mathbf{x}), \tag{4.8}$$

where m denotes the number of image pixels. This quadratic loss function leads to the Bayes estimator

$$\hat{\mathbf{x}} = \arg\min_{\tilde{\mathbf{x}}} \int \frac{1}{m} \|\tilde{\mathbf{x}} - \mathbf{x}\|^2 p(\mathbf{x}|\mathbf{y})\, d\mathbf{x} \tag{4.9}$$

$$= \arg\min_{\tilde{\mathbf{x}}} \tilde{\mathbf{x}}^\mathsf{T}\tilde{\mathbf{x}} - 2\tilde{\mathbf{x}}^\mathsf{T}\mathbb{E}[\mathbf{x}|\mathbf{y}] \tag{4.10}$$

$$= \mathbb{E}[\mathbf{x}|\mathbf{y}] \tag{4.11}$$

being the posterior mean, which is also known as the *minimum mean squared error* (MMSE) estimator. Unfortunately, computing the MMSE estimate and other distributional properties is typically more difficult as compared to MAP estimation.

We discussed in Section 2.6.2 that the PSNR is most commonly used to assess the quality of the predicted image, which we define as

$$\mathrm{PSNR}(\tilde{\mathbf{x}}, \mathbf{x}) = 10 \log_{10}\left(\frac{R^2}{\frac{1}{m}\|\tilde{\mathbf{x}} - \mathbf{x}\|^2}\right) = 20 \log_{10}\left(\frac{R\sqrt{m}}{\|\tilde{\mathbf{x}} - \mathbf{x}\|}\right), \tag{4.12}$$

where R denotes the maximum intensity level of a pixel. Using the (negative) PSNR as the loss function, *i.e.* $\Delta(\tilde{\mathbf{x}}, \mathbf{x}) = -\mathrm{PSNR}(\tilde{\mathbf{x}}, \mathbf{x})$, the associated Bayes estimator

$$\hat{\mathbf{x}} = \arg\min_{\tilde{\mathbf{x}}} \int -\mathrm{PSNR}(\tilde{\mathbf{x}}, \mathbf{x}) p(\mathbf{x}|\mathbf{y})\, d\mathbf{x} \tag{4.13}$$

$$= \arg\min_{\tilde{\mathbf{x}}} \int \log\left(\|\tilde{\mathbf{x}} - \mathbf{x}\|^2\right) p(\mathbf{x}|\mathbf{y})\, d\mathbf{x} \tag{4.14}$$

$$\approx \arg\min_{\tilde{\mathbf{x}}} \log \int \|\tilde{\mathbf{x}} - \mathbf{x}\|^2 p(\mathbf{x}|\mathbf{y})\, d\mathbf{x} \tag{4.15}$$

$$= \mathbb{E}[\mathbf{x}|\mathbf{y}] \tag{4.16}$$

can be approximated with the MMSE estimator by upper-bounding the integral in Eq. (4.14) via Jensen's inequality [Jensen, 1906].

SAMPLE-BASED APPROXIMATION Since trying to exactly compute Bayes estimators is intractable here (except for MAP estimation), we approximate the expectations with samples drawn from the posterior distribution. Recall from Section 2.3.1 that the expected value of a function f

$$\mathbb{E}[f(\mathbf{x})|\mathbf{y}] = \int f(\mathbf{x}) p(\mathbf{x}|\mathbf{y})\, d\mathbf{x} \approx \frac{1}{T}\sum_{t=1}^{T} f\left(\mathbf{x}^{(t)}\right) \tag{4.17}$$

can be approximated with a set of samples $\{\mathbf{x}^{(t)}\}_{t=1}^T \sim p(\mathbf{x}|\mathbf{y})$. In particular, the MMSE estimate thus simply corresponds to the sample average with $f(\mathbf{x}) = \mathbf{x}$. For HQ models as used here, we can obtain such a set of samples with the Gibbs sampler in Alg. 3.4, which yields $\{(\mathbf{x}^{(t)}, \mathbf{z}^{(t)})\}_{t=1}^T \sim p(\mathbf{x}, \mathbf{z}|\mathbf{y})$ from the augmented posterior by making use of the conditional distributions $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$ and $p(\mathbf{z}|\mathbf{x}, \mathbf{y})$.

RAO-BLACKWELLIZATION Additionally, we may be able to use a more efficient *Rao-Blackwellized* (RB) estimator. Using the augmented HQ posterior $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$, the Bayes estimator

Gelfand and Smith [1990] have already advocated to use a Rao-Blackwellized estimator in a similar scenario to ours.

$$\hat{\mathbf{x}} = \arg\min_{\tilde{\mathbf{x}}} \iint \Delta(\tilde{\mathbf{x}}, \mathbf{x}) p(\mathbf{x}, \mathbf{z}|\mathbf{y}) \, d\mathbf{x} \, d\mathbf{z} \tag{4.18}$$

$$= \arg\min_{\tilde{\mathbf{x}}} \int p(\mathbf{z}|\mathbf{y}) \int \Delta(\tilde{\mathbf{x}}, \mathbf{x}) p(\mathbf{x}|\mathbf{z}, \mathbf{y}) \, d\mathbf{x} \, d\mathbf{z} \tag{4.19}$$

$$= \arg\min_{\tilde{\mathbf{x}}} \int p(\mathbf{z}|\mathbf{y}) \mathbb{E}\left[\Delta(\tilde{\mathbf{x}}, \mathbf{x}) | \mathbf{z}, \mathbf{y}\right] d\mathbf{z} \tag{4.20}$$

$$\approx \arg\min_{\tilde{\mathbf{x}}} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\Delta(\tilde{\mathbf{x}}, \mathbf{x}) | \mathbf{z}^{(t)}, \mathbf{y}\right] \tag{4.21}$$

can be better approximated with lower variance if the expected loss w.r.t. $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$ can be computed analytically. Furthermore, assuming the loss function to be the (mean) squared error, this simplifies to averaging of conditional expectations of $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$:

$$\hat{\mathbf{x}} \approx \arg\min_{\tilde{\mathbf{x}}} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}\left[\|\tilde{\mathbf{x}} - \mathbf{x}\|^2 | \mathbf{z}^{(t)}, \mathbf{y}\right] \tag{4.22}$$

$$= \arg\min_{\tilde{\mathbf{x}}} \frac{1}{T} \sum_{t=1}^{T} \tilde{\mathbf{x}}^{\mathsf{T}} \tilde{\mathbf{x}} - 2\tilde{\mathbf{x}}^{\mathsf{T}} \mathbb{E}[\mathbf{x} | \mathbf{z}^{(t)}, \mathbf{y}] \tag{4.23}$$

$$= \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[\mathbf{x} | \mathbf{z}^{(t)}, \mathbf{y}]. \tag{4.24}$$

Fortunately, $p(\mathbf{x}|\mathbf{z}, \mathbf{y})$ here is a Gaussian distribution and its conditional expectation is thus tractable and can be computed by solving a system of linear equations (Section 3.5).

4.4 BAYESIAN RESTORATION USING SAMPLING

As is common, we are mainly interested in evaluating restored images in terms of PSNR, hence we choose to use MMSE estimation as a convenient approximation of the Bayes estimator for a PSNR-based loss. Importantly, employing a Bayes estimator requires an accurate posterior distribution, which here especially hinges on a good image prior. To that end, we adopt the learned high-order MRF priors from Schmidt et al. [2010], which have been shown to exhibit good generative properties. In particular, Schmidt et al. [2010] found that MMSE estimation with their learned priors leads to superior image denoising results as compared to MAP estimation. In agreement with Bayesian decision theory, they demonstrated that MMSE estimates yield a high correlation between the image restoration performance and the generative quality of the model. Another advantage of using an accurate posterior and a suitable estimator is that we do not have to use a corrective regularization parameter (*cf.* Eq. 4.7) to balance prior and likelihood;

Schmidt et al. [2010] also used PSNR to evaluate the quality of restored images.

this is especially advantageous when the noise level is not known. Finally, Schmidt et al. [2011] have already shown that MMSE estimation with the priors of Schmidt et al. [2010] leads to excellent non-blind deblurring results. We adopt and extend the sampling-based inference approach of Schmidt et al. [2011], which we describe in the following before we discuss our extensions in Sections 4.5 and 4.6.

HALF-QUADRATIC INFERENCE Combining likelihood (Eq. 4.1) and augmented prior (Eq. 4.5), we obtain the augmented posterior

$$p(\mathbf{x},\mathbf{z}|\mathbf{y},\mathbf{K},\sigma) \propto p(\mathbf{y}|\mathbf{x},\mathbf{K},\sigma) \cdot p(\mathbf{x},\mathbf{z}) \tag{4.25}$$

for the image restoration problem. To employ the Gibbs sampler from Alg. 3.4 for sampling-based inference, we first need to specify the necessary conditional distributions. First,

$$p(\mathbf{z}|\mathbf{x},\mathbf{y},\mathbf{K},\sigma) \propto p(\mathbf{z}|\mathbf{x}) \propto \prod_{c\in\mathcal{C}} \prod_{i=1}^{N} \alpha_{iz_{ic}} \mathcal{N}(\mathbf{f}_i^{\mathsf{T}}\mathbf{x}_{(c)}; 0, \eta_{iz_{ic}}^2) \tag{4.26}$$

is generally not affected by the likelihood term and decomposes since the latent variables do not interact (*cf.* Section 3.3.2.2). As a result, each z_{ic} adheres to a univariate discrete distribution and can be sampled very easily. Since we use a multiplicative HQ form, the conditional distribution

$$\begin{aligned} p(\mathbf{x}|\mathbf{z},\mathbf{y},\mathbf{K},\sigma) &\propto \mathcal{N}\Big(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2\mathbf{I}\Big) \cdot \mathcal{N}\Big(\mathbf{x}; \mathbf{0}, \big(\textstyle\sum_{i=1}^{N} \mathbf{F}_i^{\mathsf{T}}\mathbf{Z}_i\mathbf{F}_i\big)^{-1}\Big) \\ &\propto \mathcal{N}\Big(\mathbf{x}; \mathbf{\Omega}_{\mathbf{z}}^{-1}\mathbf{K}^{\mathsf{T}}\frac{\mathbf{y}}{\sigma^2}, \mathbf{\Omega}_{\mathbf{z}}^{-1}\Big) \end{aligned} \tag{4.27}$$

is jointly Gaussian in $\mathbf{x}$ with precision matrix

$$\mathbf{\Omega}_{\mathbf{z}} = \sum_{i=1}^{N} \mathbf{F}_i^{\mathsf{T}}\mathbf{Z}_i\mathbf{F}_i + \frac{1}{\sigma^2}\mathbf{K}^{\mathsf{T}}\mathbf{K}, \tag{4.28}$$

which depends on latent variables $\mathbf{z}$ in a similar way to Eq. (3.54). The only difference here is that each z_{ic} indicates one of the Gaussian mixture components (Eq. 4.4), resulting in the diagonal matrices $\mathbf{Z}_i = \mathcal{D}_{\mathcal{C}}\{\eta_{iz_{ic}}^{-2}\}$ [*cf.* Schmidt et al., 2010]. Since Eq. (4.27) is Gaussian, we can sample from it by solving a sparse system of linear equations (Section 3.5.3).

The matrices $\mathbf{F}_i$ are defined in the same way as used in Eq. (3.54).

We perform posterior sampling according to the Gibbs sampler of Alg. 3.4, which alternates between sampling from Eqs. (4.26) and (4.27). After a suitable amount of burn-in iterations, we obtain a sequence of samples $\{\{\mathbf{z}^{(1)},\mathbf{x}^{(1)}\},\ldots,\{\mathbf{z}^{(T)},\mathbf{x}^{(T)}\}\}$. As explained earlier, the MMSE estimate of $\mathbf{x}$ can now be approximated by simply averaging the samples $\mathbf{x}^{(t)}$. However, we also discussed the alternative possibility of using a Rao-Blackwellized (RB) MMSE estimator

$$\hat{\mathbf{x}}_{\mathrm{RB}} \approx \frac{1}{T}\textstyle\sum_{t=1}^{T} \mathbf{\Omega}_{\mathbf{z}^{(t)}}^{-1}\mathbf{K}^{\mathsf{T}}\frac{\mathbf{y}}{\sigma^2}, \tag{4.29}$$

which averages the conditional expectations from Eq. (4.27). We make use of Rao-Blackwellization for the experiments in Section 4.7, since Schmidt et al. [2011] found that it enabled them to draw fewer samples to satisfy their convergence criteria (which we also adopt). However, we remark that using an RB-MMSE estimator requires to solve one additional system of linear equations per Gibbs iterations (*cf.* Section 3.5.3). Hence, while overall fewer Gibbs iterations are necessary, each iteration is computationally more expensive.

4.5 NOISE ESTIMATION

Most MAP-based approaches in low-level vision rely on the choice of a regularization parameter λ that calibrates the influence of prior and likelihood on the posterior (*cf.* Eq. 4.7). This parameter is dependent on the noise level and must in practice be determined in an off-line training step. Nonetheless, even after training the regularization parameter, the user must still provide a noise level estimate, which can significantly affect the application performance when selected incorrectly. Fig. 4.1 shows how the image deblurring performance depends on the chosen noise level for a selection of deblurring methods, and illustrates that a reliable noise estimate is crucial for optimal deblurring performance. One of the properties of the MMSE-based restoration approach [Schmidt et al., 2010, 2011] from Section 4.4 is that it does not require off-line training of a regularization parameter. In contrast, all other approaches shown in Fig. 4.1 require such a procedure.

In the following, we further extend the framework of Section 4.4 with integrating noise estimation. Specifically, we adopt a Bayesian approach and treat σ as an unobserved random variable, which we (approximately) integrate out:

$$p(\mathbf{x}|\mathbf{y},\mathbf{K}) = \int p(\mathbf{x},\sigma|\mathbf{y},\mathbf{K})\, d\sigma. \tag{4.30}$$

To that end we incorporate the noise σ as a new variable in an extended joint distribution $p(\mathbf{x},\mathbf{z},\sigma|\mathbf{y},\mathbf{K})$. Since the input image and the blur kernel provide sufficient constraints in practice, we assume a uniform prior on σ, *i.e.* $p(\sigma) = \text{const}$. To estimate the integral from Eq. (4.30), we extend the Gibbs sampler from Section 4.4 by another step for sampling the conditional distribution $p(\sigma|\mathbf{x},\mathbf{z},\mathbf{y},\mathbf{K})$, which is a Gamma distribution $\mathcal{G}(x;a,b) = \frac{x^{a-1}e^{-x/b}}{b^a\,\Gamma(a)}$ on the inverse noise variance [Fergus et al., 2006]:

$$\begin{aligned} p(\sigma|\mathbf{x},\mathbf{z},\mathbf{y},\mathbf{K}) &\propto \mathcal{N}(\mathbf{y};\mathbf{K}\mathbf{x},\sigma^2\mathbf{I}) && (4.31)\\ &\propto \sigma^{-n}\exp\left(-\frac{\|\mathbf{y}-\mathbf{K}\mathbf{x}\|^2}{2\sigma^2}\right) && (4.32)\\ &\propto \mathcal{G}\left(\frac{1}{\sigma^2};\frac{n}{2}+1,\frac{2}{\|\mathbf{y}-\mathbf{K}\mathbf{x}\|^2}\right). && (4.33) \end{aligned}$$

Gibbs sampling proceeds by sampling σ, $\mathbf{z}$ and $\mathbf{x}$ alternatingly, yielding a sequence of samples $\{\{\mathbf{z}^{(t)}, \mathbf{x}^{(t)}, \sigma^{(t)}\}\}_{t=1}^{T}$. Hence, we can obtain an MMSE estimate of the deblurred image $\mathbf{x}$ without knowledge of the noise level as

$$\hat{\mathbf{x}} = \arg\min_{\tilde{\mathbf{x}}} \iint \|\tilde{\mathbf{x}} - \mathbf{x}\|^2 p(\mathbf{x}, \sigma|\mathbf{y}, \mathbf{K})\, d\mathbf{x}\, d\sigma = \mathbb{E}[\mathbf{x}|\mathbf{y}, \mathbf{K}], \tag{4.34}$$

again by simply averaging samples $\mathbf{x}^{(t)}$ from the posterior. The Rao-Blackwellized MMSE estimator of Eq. (4.29) only needs to be slightly adjusted by replacing σ with the current sample $\sigma^{(t)}$.

NOISE ESTIMATOR Besides the deblurred image, we may also be interested in an estimate of the noise level σ itself. Following Zoran and Weiss [2009], we will evaluate the quality of a noise level estimate $\hat{\sigma}$ based on the *relative absolute error*

$$\Delta(\hat{\sigma}, \sigma_{\mathrm{GT}}) = \frac{|\hat{\sigma} - \sigma_{\mathrm{GT}}|}{\sigma_{\mathrm{GT}}} = |\hat{\sigma}/\sigma_{\mathrm{GT}} - 1|, \tag{4.35}$$

where $\sigma_{\mathrm{GT}} > 0$ denotes the correct noise standard deviation. Given this error (loss) function, the corresponding Bayes estimator

$$\hat{\sigma} = \arg\min_{\tilde{\sigma}} \iint \Delta(\tilde{\sigma}, \sigma) p(\mathbf{x}, \sigma|\mathbf{y}, \mathbf{K})\, d\mathbf{x}\, d\sigma \approx \arg\min_{\tilde{\sigma}} \frac{1}{T} \sum_{t=1}^{T} |\tilde{\sigma}/\sigma^{(t)} - 1| \tag{4.36}$$

does not have a closed-form solution, but can easily be approximated by solving a convex one-dimensional optimization problem based on the obtained set of samples. This is how we estimate the noise standard deviation in our experiments (Section 4.7).

In contrast to the estimation framework of Fergus et al. [2006], our sampling-based method allows to model sensor noise in the spatial domain. Moreover, the fundamental difference to standard MAP approaches is that our noise estimation process is a fully automatic, built-in procedure of the deblurring algorithm, which arises naturally by treating the noise standard deviation as a variable of the posterior. This has the advantage that the noise estimation procedure is specialized to the image restoration problem at hand, here deblurring.

We would also like to point out the degenerate case of identity blur $\mathbf{K} = \mathbf{I}$, in which case samples drawn from $p(\mathbf{x}, \mathbf{z}, \sigma|\mathbf{y}, \mathbf{K})$ can effectively be used for blind denoising with simultaneous noise estimation.

4.6 PARAMETRIC BLUR ESTIMATION

Assuming spatially uniform blur, *i. e.* $\mathbf{K}\mathbf{x} \equiv \mathbf{k} \otimes \mathbf{x}$ corresponds to convolution, we can in principle treat the blur kernel $\mathbf{k}$ as an additional unobserved random vector and proceed in a similar way as we have done for the unobserved noise level. Unfortunately, we empirically find this not to work well, in part possibly due to the larger state space that the Gibbs sampler has to explore. However, this might be successful when we have strong prior knowledge about the blur. Taking this idea a step further, we can restrict the blur estimation to specific kinds of (parametric) blur that only depend on very few parameters.

The blur kernel is assumed to be odd-sized, i. e. *w is an odd number.*

GAUSSIAN BLUR As a first example, we will address the removal of *Gaussian blur*, which we parameterize with a single bandwidth parameter ν. Concretely, we assume a blur kernel of size $w \times w$, which we denote by the vector $\mathbf{k} \in \mathbb{R}^{w^2}$, whose normalized entries are defined as

$$k_{ij} = \frac{1}{S} \exp\left(-\frac{1}{2\nu^2} \left((i-c)^2 + (j-c)^2 \right) \right) \tag{4.37}$$

where the indices i, j correspond to the (2D) location of the respective entry, $c = (w-1)/2$ denotes the location of the central pixel, and S is a normalization constant such that $\sum_{i,j} k_{ij} = 1$.

Assuming a uniform prior on ν and denoting convolution as $\mathbf{k} \otimes \mathbf{x} \equiv \mathbf{K}\mathbf{x} = \mathcal{C}_{\mathbf{x}}\mathbf{k}$ with matrix $\mathcal{C}_{\mathbf{x}}$ derived from all overlapping $w \times w$-sized patches of $\mathbf{x}$, we can write the conditional distribution of ν as

$$\begin{aligned} p(\nu|\mathbf{x}, \mathbf{z}, \mathbf{y}, \sigma) &\propto \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2 \mathbf{I}) \\ &\propto \exp\left(\frac{1}{\sigma^2} \left(-\frac{1}{2} \mathbf{k}^\mathsf{T} \mathcal{C}_{\mathbf{x}}^\mathsf{T} \mathcal{C}_{\mathbf{x}} \mathbf{k} + \mathbf{k}^\mathsf{T} \mathcal{C}_{\mathbf{x}}^\mathsf{T} \mathbf{y} \right) \right). \end{aligned} \tag{4.38}$$

Although $p(\nu|\mathbf{x}, \mathbf{z}, \mathbf{y}, \sigma)$ does not seem to correspond to a well-known univariate distribution, we can efficiently evaluate it (in unnormalized form) based on matrix-vector products whose size only depends on that of the blur kernel, but not the image. Hence, we can simply discretize the distribution over the domain of ν. Note that $\mathcal{C}_{\mathbf{x}}^\mathsf{T} \mathcal{C}_{\mathbf{x}}$ and $\mathcal{C}_{\mathbf{x}}^\mathsf{T} \mathbf{y}$ can be cached, since they do not depend on ν.

Replacing $\mathbf{K}$ with the new unobserved random variable ν (since $\mathbf{K}$ is fully defined by ν), we now work with the joint distribution $p(\mathbf{x}, \mathbf{z}, \sigma, \nu|\mathbf{y})$ and therefore extend the Gibbs sampler from Section 4.5 with an additional step to also sample from Eq. (4.38). Hence, we obtain the extended sequence of samples $\{\{\mathbf{z}^{(t)}, \mathbf{x}^{(t)}, \sigma^{(t)}, \nu^{(t)}\}\}_{t=1}^{T}$.

Estimation of the deblurred image $\mathbf{x}$ is adjusted in the same way as we have done when integrating noise estimation (Section 4.5). As for predicting the bandwidth parameter ν of the Gaussian blur, we use MMSE estimation, thus assuming a squared loss. However, we find that using the (relative) absolute error as loss function leads to virtually the same estimates in our experiments (Section 4.7).

LINEAR MOTION BLUR As a second example, we consider linear motion deconvolution, where the blur kernel $\mathbf{k} = \text{line}^1(l, \theta)$ corresponds to a straight line given by length l (in pixels) and angle θ (in

---

[1] We use the MATLAB function `fspecial('motion',`l`,`θ`)`.

degrees). Convolving an image with such a kernel (approximately) corresponds to a linear motion of the *camera* along the image plane (*cf.* Fig. 1.3(a)).

Our deconvolution approach is essentially the same as for removal of Gaussian blur as explained above. In particular, we use uniform priors on blur parameters l and θ and obtain and evaluate the conditional distributions $p(l|\mathbf{x},\mathbf{z},\mathbf{y},\sigma,\theta)$ and $p(\theta|\mathbf{x},\mathbf{z},\mathbf{y},\sigma,l)$ in analogy to Eq. (4.38). We end up with the joint distribution $p(\mathbf{x},\mathbf{z},\sigma,l,\theta|\mathbf{y})$ and extend the Gibbs sampler from Section 4.5 with two additional steps to sample from $p(l|\mathbf{x},\mathbf{z},\mathbf{y},\sigma,\theta)$ and $p(\theta|\mathbf{x},\mathbf{z},\mathbf{y},\sigma,l)$, eventually yielding a sequence of samples $\{\{\mathbf{z}^{(t)},\mathbf{x}^{(t)},\sigma^{(t)},l^{(t)},\theta^{(t)}\}\}_{t=1}^{T}$. As before, estimation of the deblurred image $\mathbf{x}$ is adjusted in analogy to Section 4.5, where blur parameters l,θ are obtained via the MMSE.

The qualitative results in Section 4.7 demonstrate the feasibility of our approach to estimate Gaussian and linear motion blur, and thus show that other quantities besides the noise level can be estimated.

4.7 EXPERIMENTS

MATLAB code is available (in part) on our webpage.

We demonstrate the benefits of our approach with several tasks. Most importantly, we show that the quality of the restored images deteriorates only slightly when not relying on a known noise level; to that end, we compare our results in the context of image denoising [Schmidt et al., 2010] and non-blind deblurring [Schmidt et al., 2011]. Furthermore, since we also obtain a noise estimate with our approach, we compare this against the dedicated noise estimation approach of Zoran and Weiss [2009]. Finally, we demonstrate promising results for blind deconvolution with noise and blur estimation for two kinds of parametric blur: Gaussian blur and linear camera motion.

We use the learned image priors from Schmidt et al. [2010]; in particular, we carry out all experiments with a pairwise MRF and a high-order 3×3 FoE prior. Besides the good quality of these priors, this also allows us to directly compare with previous results for the case when the noise level is known [Schmidt et al., 2010, 2011].

BLIND DENOISING WITH NOISE ESTIMATION We begin with image denoising, *i. e.* the matrix $\mathbf{K}$ equals the identity matrix in Eq. (4.1). In this case, our method enables us to perform blind denoising (unknown σ) with integrated noise estimation. Based on the approach described in Sections 4.4 and 4.5 (with $\mathbf{K} = \mathbf{I}$), we perform a series of experiments on 68 images used in [Roth and Black, 2009]. The results are summarized in Tab. 4.1 using PSNR[2] and also structural similar-

Although our estimator is chosen for PSNR, it also works well for SSIM.

---

[2] To facilitate comparisons to the results from Schmidt et al. [2010], the PSNR values in Table 4.1 are obtained using the slightly different definition $\mathrm{PSNR}_{\mathrm{SD}}(\tilde{\mathbf{x}},\mathbf{x}) = 20\log_{10}\big((R\sqrt{m-1})/\|\mathbf{d} - \frac{1}{m}\mathbf{1}^{\mathsf{T}}\mathbf{d}\|\big)$ with $\mathbf{d} = \tilde{\mathbf{x}} - \mathbf{x}$ (*cf.* Eq. 2.47). This yields very similar results compared to our definition in Eq. (4.12) if $\frac{1}{m}\mathbf{1}^{\mathsf{T}}\mathbf{d} \approx 0$, which is mostly the case (*cf.* Section 2.6.2.2). In practice, (average) results may differ at most 0.05dB.

Approach	Estimate $\hat{\sigma}$		PSNR	SSIM
	avg.	$\langle\epsilon\rangle$	avg. dB	avg.
5×5 FoE (MAP) [Roth and Black, 2009]	GT	—	27.44	0.746
5×5 FoE (MAP) [Samuel and Tappen, 2009]	GT	—	27.86	0.776
Pairwise MRF (MMSE) [Schmidt et al., 2010]	GT	—	27.54	0.758
3×3 FoE (MMSE) [Schmidt et al., 2010]	GT	—	**27.95**	**0.788**
Zoran and Weiss [2009]	23.16	8.8%	—	—
Ours, pairwise MRF (MMSE)	22.81	10.1%	27.16	0.733
Ours, 3×3 FoE (MMSE)	24.21	**5.8%**	27.88	0.783

Table 4.1: **Average denoising results and noise estimates** $\hat{\sigma}$ for 68 test images and $\sigma = 25$ (partly reproduced from Schmidt et al. [2010]). The average relative error is shown as $\langle\epsilon\rangle = \frac{1}{68}\sum_{k=1}^{68}|\hat{\sigma}_k - \sigma|/\sigma$; GT denotes that the true value for σ was used.

ity (SSIM) [Wang et al., 2004]. Most importantly, we find that despite unknown noise level our average results are only slightly worse than the non-blind denoising results with a 3×3 FoE reported by Schmidt et al. [2010] (27.88dB vs. 27.95dB). Nonetheless, the performance on individual images can significantly differ. Moreover, if we use our approach to perform noise estimation, we obtain results that are superior to those of Zoran and Weiss [2009]. This is an interesting result due to the conceptual simplicity of our approach: It is solely guided by a noise model and a natural image prior.

The performance in case of a pairwise MRF drops behind the non-blind setting more significantly ($\sim$0.4dB worse). Moreover, noise estimates are also slightly inferior to Zoran and Weiss [2009] in this case. It is interesting to note that we observe neither effect in case of deblurring (see below). Finally, Tab. 4.1 also shows that our MMSE-based approach with integrated noise estimation outperforms the MAP-based approaches with 5×5 FoEs of Roth and Black [2009] and Samuel and Tappen [2009] despite the fact that they rely on knowledge of the noise parameter σ.

NON-BLIND DEBLURRING WITH NOISE ESTIMATION We exactly followed the experimental approach of Schmidt et al. [2011] to facilitate a direct comparison with our method for the task of non-blind deblurring. To that end, we tested our method on 64 synthetically blurred test images of size 128×128 pixels, which are corrupted with Gaussian noise of three different noise levels ($\sigma = 2.55, 7.65, 12.75$), yielding three test sets overall. Note that to mimic a somewhat more realistic scenario, the test images are 8-bit quantized and a slightly perturbed version of the true blur kernel is used for deblurring [*cf.* Schmidt et al., 2011].

Model	Performance loss in PSNR (dB) / SSIM					
	$\sigma = 2.55$		$\sigma = 7.65$		$\sigma = 12.75$	
Pairwise MRF (MMSE)	0.07	0.003	0.05	0.004	0.03	0.004
3×3 FoE (MMSE)	0.04	0.004	0.04	0.004	0.03	0.003

Table 4.2: **Comparison of average deblurring results** for 64 test images and three noise levels between the approach of Schmidt et al. [2011] and ours. While Schmidt et al. [2011] used the GT noise level σ, we did not assume knowledge of the true noise level and instead employed our integrated noise estimation approach (Section 4.5). Note that the average loss in image quality is minor for all tested noise levels.

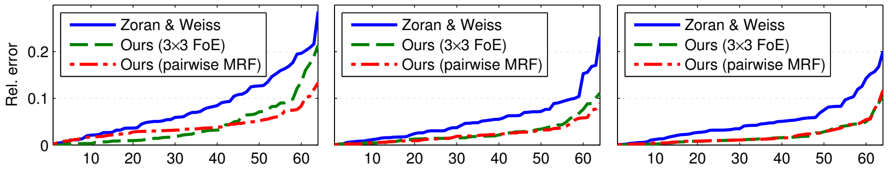
As for image denoising, we also find for non-blind deblurring that integrated noise estimation performs almost identically to deblurring with the (usually unknown) ground truth noise level (Table 4.2). Figs. 4.3 and 4.4 show qualitative comparisons of deblurred images between our method and some common competing approaches that had also been considered in [Schmidt et al., 2011]. In particular, this includes the popular non-blind deblurring approaches of Levin et al. [2007] and Krishnan and Fergus [2009], and the learned FoE prior of Roth and Black [2009]; inference for all of these methods is carried out via half-quadratic MAP estimation. The well-known Lucy-Richardson method [Lucy, 1974; Richardson, 1972] is also included as a baseline.

We also evaluated the noise estimation performance itself by comparing with the approach of Zoran and Weiss [2009], which is one of the most competitive techniques in this area. We report results for the 64 blurred images and three noise levels in Tab. 4.3; a visual comparison is given in Fig. 4.2. We find that our estimates are substantially better than those of Zoran and Weiss [2009] in terms of the average relative estimation error. This holds for the pairwise MRF and high-order FoE model as well as all noise levels, particularly the large ones, and demonstrates the advantage of having a noise estimation procedure that is specifically adapted to the problem at hand (here, image deblurring).

BLIND DECONVOLUTION WITH NOISE ESTIMATION Finally, we show some qualitative results for blind deconvolution with parametric blur estimation (concretely Gaussian and linear motion blur) to demonstrate the applicability of our approach to other quantities besides the noise level. Additionally, we also do not assume the noise level to be known. Overall, this makes our method very appealing, since there are no parameters to tune and the restoration process is solely based on a learned natural image prior and the modeling assumptions for noise and blur.

Approach	$\sigma = 2.55$			$\sigma = 7.65$			$\sigma = 12.75$		
	avg.	std.	$\langle \epsilon \rangle$	avg.	std.	$\langle \epsilon \rangle$	avg.	std.	$\langle \epsilon \rangle$
Zoran and Weiss [2009]	2.52	0.27	8.21%	7.44	0.50	5.34%	12.41	0.80	5.18%
Ours (pairwise MRF)	2.53	0.12	**3.95**%	7.67	0.23	**2.25**%	12.81	0.38	2.02%
Ours (3×3 FoE)	2.64	0.14	4.19%	7.78	0.24	2.52%	12.86	0.35	**1.95**%

Table 4.3: **Noise estimation results** for the 64 blurred test images and three noise levels, comparing our method with [Zoran and Weiss, 2009]. The average relative error is denoted by $\langle \epsilon \rangle = \frac{1}{64} \sum_{k=1}^{64} |\hat{\sigma}_k - \sigma| / \sigma$.



(a) $\sigma = 2.55$ (b) $\sigma = 7.65$ (c) $\sigma = 12.75$

Figure 4.2: **Relative noise estimation errors** $|\hat{\sigma} - \sigma| / \sigma$ for the 64 blurred test images and three noise levels, comparing our method with [Zoran and Weiss, 2009]. Each curve is sorted *separately* and does not indicate a performance comparison between the methods for a given image.

At the end of the restoration process, we obtain estimates for the restored image $\hat{\mathbf{x}}$, the Gaussian noise strength $\hat{\sigma}$, and the blur parameters ($\hat{\nu}$ for Gaussian blur; $\hat{l}, \hat{\theta}$ for linear motion blur; *cf.* Section 4.6). Figs. 4.5 and 4.6 each show results for two synthetically corrupted example images, one with relatively little blur and some noise, the other with stronger blur and little noise. A first observation is that for both types of blur and MRF models, the noise level estimates $\hat{\sigma}$ are generally quite good. The blur strength seems to be somewhat underestimated in case of Gaussian deconvolution (Fig. 4.5), *i.e.* $\hat{\nu}$ is below the true value; nevertheless, the restored images look reasonable, especially the pairwise MRF yields sharp images. Although the estimates for linear motion blur are generally good (Fig. 4.5), the restored images can exhibit ringing artifacts in case of strong blur (Fig. 4.5(e-f)); overall, the restored images are much sharper and improve on the observed blurred ones in terms of PSNR and SSIM.

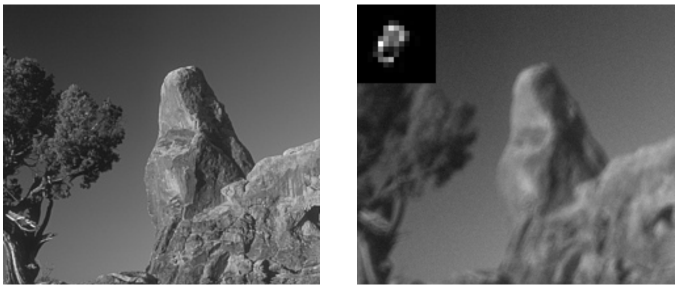
As a technical detail, we remark that the conditional distributions of the blur parameters can be very peaked when the noise parameter σ is small. Hence, at the beginning of the sampling process, we first gradually decrease σ starting from a large value, before we begin updating it normally by sampling from its respective distribution (Eq. 4.33). Note that employing an annealing schedule for the noise level is actually quite common in the context of blind deblurring [*cf.* Levin et al., 2011; Wipf and Zhang, 2014].

4.8 SUMMARY

We proposed a Bayesian framework for image restoration that can effectively handle unobserved variables of the likelihood model. A key component that made our approach practical is half-quadratic inference via block Gibbs sampling. In particular, it is necessary to use HQ augmentation of the integral type since we need to draw samples from the posterior distribution. Furthermore, using the multiplicative form affords fast mixing of the sampler, such that a few hundred iterations are sufficient to yield good results for approximate Bayesian inference. Note that our proposed additional sampling steps for the latent likelihood variables are negligible in terms of overall runtime.

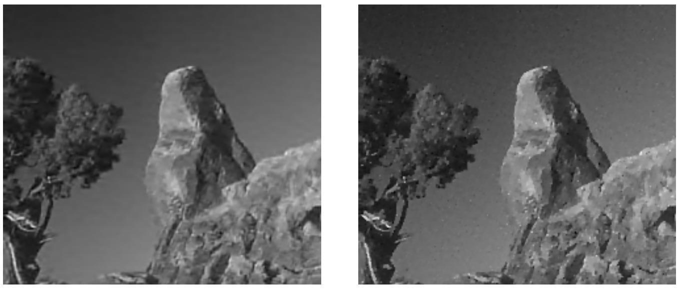
Our experiments in the context of image denoising and deblurring show that even without relying on a known noise level the restoration quality differs negligibly on average to the case where the noise level is known. Furthermore, the quality of the obtained noise estimates is competitive to dedicated noise estimation methods. We also showed promising results for blind deconvolution with Gaussian and linear motion blur, where we estimated the noise and blur besides the restored image.

Moreover, our approach of integrated estimation of nuisance parameters readily extends to other problems. Since the publication of [Schmidt et al., 2011], our sampling-based inference framework with integrated noise estimation has been extended to super-resolution [Zhang et al., 2012], image separation [Zhang and Zhang, 2012], and depth estimation with spatially-varying noise [Wang et al., 2014]. Also, Zhao et al. [2013] extended our non-blind deblurring approach by combining our inference framework with the non-local range MRF of Sun and Tappen [2011]; they perform noise estimation based on the model of Shan et al. [2008].



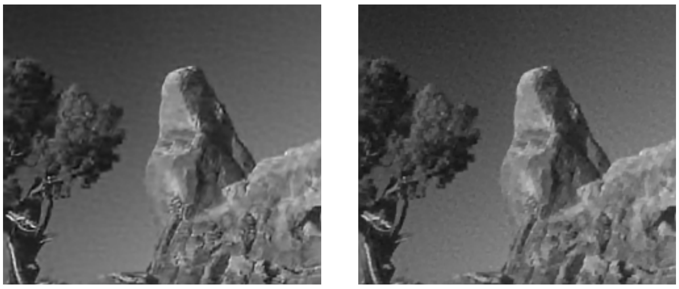
(a) Original

(b) Blurred
PSNR = 23.89dB, SSIM = 0.588



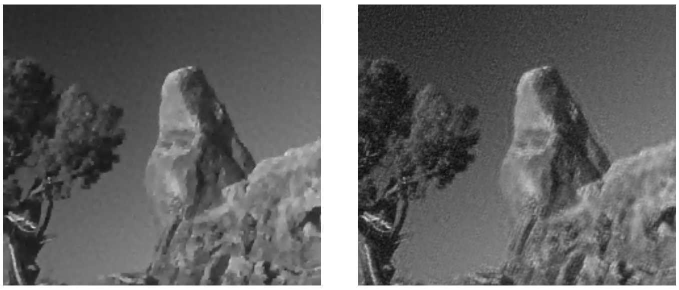
(c) Ours (3 × 3 FoE)
PSNR = 29.05dB, SSIM = 0.864

(d) Ours (pairwise MRF)
PSNR = 28.84dB, SSIM = 0.842



(e) 5 × 5 FoE (MAP)
[Roth and Black, 2009]
PSNR = 28.81dB, SSIM = 0.844

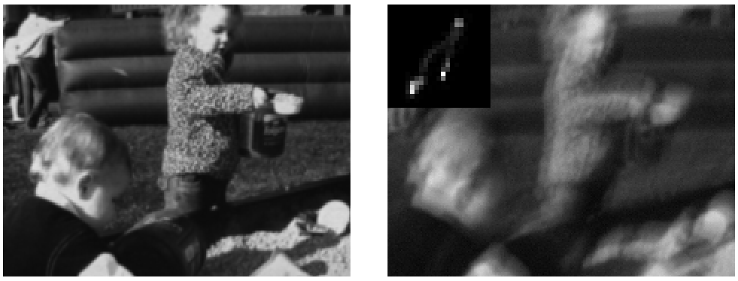
(f) 2 × 2 MRF (MAP)
[Levin et al., 2007]
PSNR = 28.54dB, SSIM = 0.826



(g) pairwise MRF (MAP)
[Krishnan and Fergus, 2009]
PSNR = 28.36dB, SSIM = 0.825

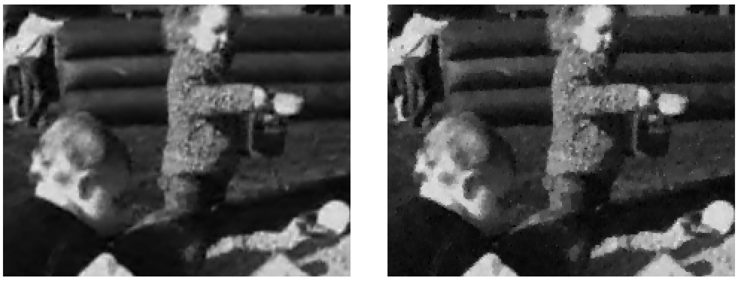
(h) Lucy-Richardson [Lucy, 1974; Richardson, 1972]
PSNR = 27.01dB, SSIM = 0.693

Figure 4.3: **Deblurring example (cropped).** Comparison of methods from Table 4.2, where all methods except ours used the ground truth noise level; we employ noise estimation (Section 4.5). The blur (kernel of size 15 × 15) is shown in the upper right corner of *(b)* (resized and scaled for better visualization). *Best viewed on screen.*



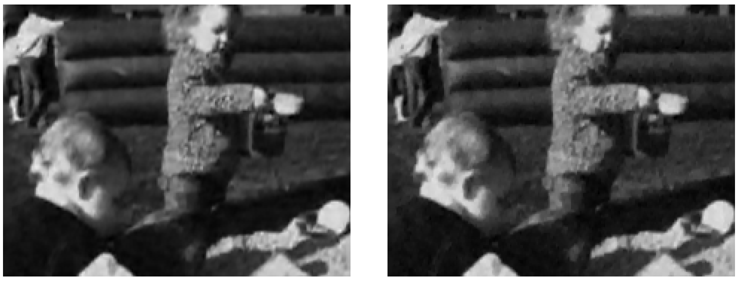
(a) Original

(b) Blurred
PSNR = 19.24dB, SSIM = 0.480



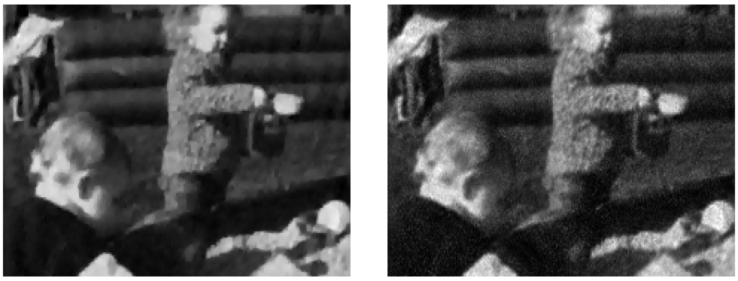
(c) Ours (3 × 3 FoE)
PSNR = 32.09dB, SSIM = 0.920

(d) Ours (pairwise MRF)
PSNR = 30.31dB, SSIM = 0.878



(e) 5 × 5 FoE (MAP)
[Roth and Black, 2009]
PSNR = 31.71dB, SSIM = 0.914

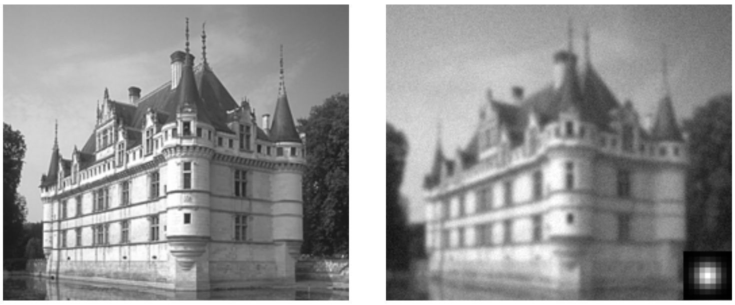
(f) 2 × 2 MRF (MAP)
[Levin et al., 2007]
PSNR = 31.33dB, SSIM = 0.898



(g) pairwise MRF (MAP)
[Krishnan and Fergus, 2009]
PSNR = 28.30dB, SSIM = 0.873

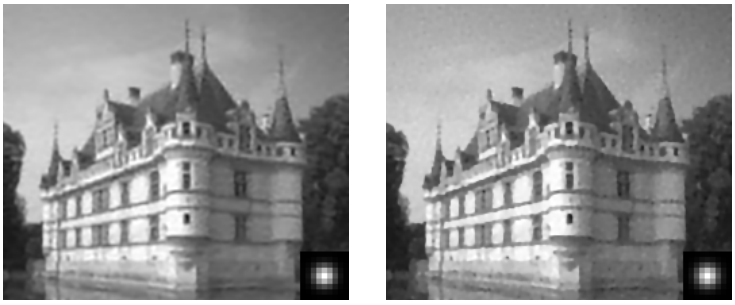
(h) Lucy-Richardson
[Lucy, 1974; Richardson, 1972]
PSNR = 26.23dB, SSIM = 0.708

Figure 4.4: **Deblurring example (cropped).** Comparison of methods from Table 4.2, where all methods except ours used the ground truth noise level; we employ noise estimation (Section 4.5). The blur (kernel of size 23 × 23) resembles typical camera shake; it is shown in the upper right corner of *(b)* (resized and scaled for better visualization). *Best viewed on screen.*



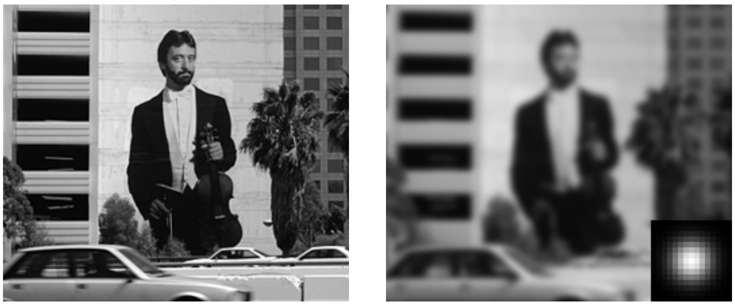
(a) Original

(b) Blurred ($\sigma = 5, \nu = 1.5$)
PSNR = 23.52dB, SSIM = 0.637



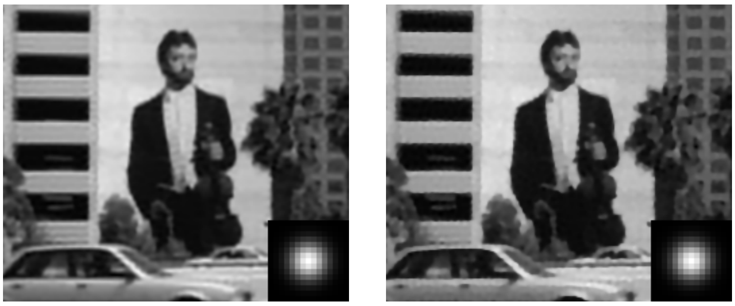
(c) 3×3 FoE ($\hat{\sigma} = 4.96, \hat{\nu} = 1.30$)
PSNR = 25.43dB, SSIM = 0.807

(d) pw. MRF ($\hat{\sigma} = 4.94, \hat{\nu} = 1.35$)
PSNR = 25.63dB, SSIM = 0.802



(e) Original

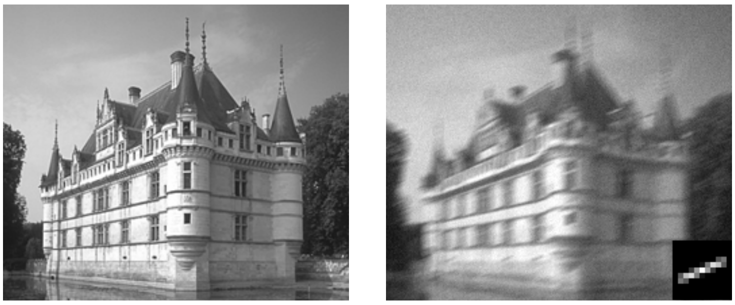
(f) Blurred ($\sigma = 0.5, \nu = 2.5$)
PSNR = 19.76dB, SSIM = 0.548



(g) 3×3 FoE ($\hat{\sigma} = 0.57, \hat{\nu} = 2.25$)
PSNR = 23.47dB, SSIM = 0.752

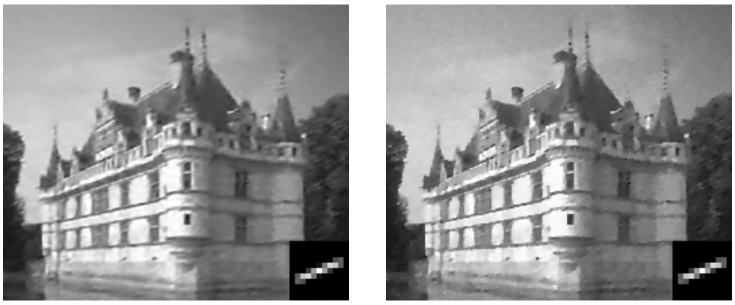
(h) pw. MRF ($\hat{\sigma} = 0.57, \hat{\nu} = 2.25$)
PSNR = 23.80dB, SSIM = 0.756

Figure 4.5: **Blind deconvolution examples (cropped)** with Gaussian blur and noise estimation using our pairwise MRF *(d,h)* and 3×3 FoE *(c,g)* for two synthetically corrupted input images *(b,f)*. Ground truth and estimated blur kernels in lower right corners are resized and scaled for visualization. *Best viewed on screen.*



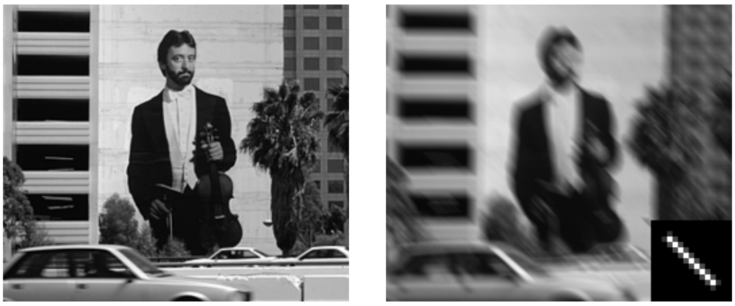
(a) Original

(b) Blurred (σ=5, l=9, θ=23°)
PSNR = 22.53dB, SSIM = 0.588



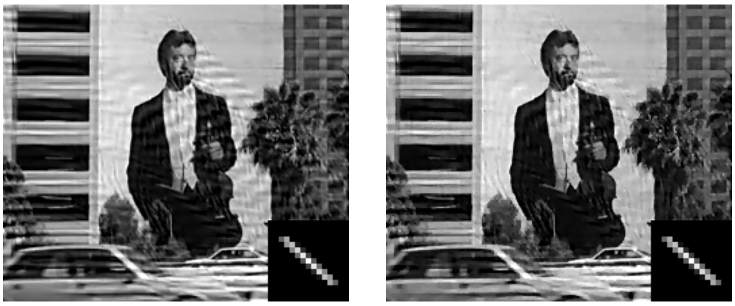
(c) 3×3 FoE ($\hat{\sigma}$=5.0, $\hat{l}$=8.8, $\hat{\theta}$=23°)
PSNR = 26.08dB, SSIM = 0.820

(d) pw. MRF ($\hat{\sigma}$=4.9, $\hat{l}$=8.9, $\hat{\theta}$=23°)
PSNR = 25.98dB, SSIM = 0.806



(e) Original

(f) Blurred (σ=0.5, l=13, θ=135°)
PSNR = 18.77dB, SSIM = 0.486



(g) $3{\times}3$ FoE ($\hat{\sigma}$=0.7, $\hat{l}$=12.7, $\hat{\theta}$=138°)
PSNR = 20.91dB, SSIM = 0.703

(h) pw.MRF ($\hat{\sigma}$=0.7, $\hat{l}$=12.8, $\hat{\theta}$=138°)
PSNR = 21.74dB, SSIM = 0.745

Figure 4.6: **Blind deconvolution examples (cropped)** with linear motion blur and noise estimation using our pairwise MRF *(d,h)* and 3×3 FoE *(c,g)* for two synthetically corrupted input images *(b,f)*. Ground truth and estimated blur kernels in lower right corners are resized and scaled for visualization. *Best viewed on screen.*

5

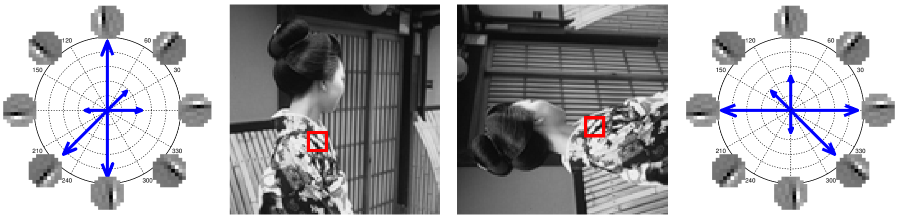
LEARNING ROTATION-AWARE FEATURES: FROM INVARIANT PRIORS TO EQUIVARIANT DESCRIPTORS

CONTENTS

5.1 Product Models & Linear Transformations 113
  5.1.1 Integrating transformation invariance 114
5.2 Learning Rotation-Invariant Image Priors 115
5.3 Learning Rotation-Aware Image Features 117
5.4 Rotation In-/Equivariant Image Descriptor 119
5.5 Experiments 123
5.6 Summary 126

Despite having been extensively studied, the problem of identifying suitable feature representations for images remains a key challenge in computer vision today. This is true in a diverse set of areas ranging from high-level tasks, such as object classification and detection [Lowe, 2004; Lazebnik et al., 2004; Dalal and Triggs, 2005; Ahonen et al., 2009; Krizhevsky et al., 2012] all the way down to problems as low-level as image restoration [Zhu and Mumford, 1997; Welling et al., 2003; Roth and Black, 2009]. Due to the diversity of areas in which feature representations are crucial, the characteristics of what makes a good feature representation also differ quite widely. One common thread in the recent literature is the increase in methods that learn suitable feature representations for specific tasks from example data [*e.g.*, Kavukcuoglu et al., 2009; Norouzi et al., 2009]. One motivation for this is that devising well-performing feature representations manually is a complex process, since it may not be very intuitive which aspects of a feature representation make it perform well in practice [Lowe, 2004; Dalal and Triggs, 2005]. Another is that customizing the feature representation to the task at hand may have significant benefits in practice.

An important shortcoming of many feature learning approaches is that they do not have the same desirable invariances or equivariances with respect to transformations of the input as do traditional hand-crafted representations. In various use cases of object detection it is, for example, reasonable to expect that an object can be detected no matter its orientation in the image. Hand-crafted feature representations [*e.g.*, Takacs et al., 2010] facilitate this by using a rotation-



(a) (b) $E = 490654$ (c) $E = 488655$ (d)

Figure 5.1: **Rotation invariance and equivariance.** *(b,c)* Current learned image priors (here [Schmidt et al., 2010]) are not rotation invariant and assign different energies E depending on the image orientation. We address this issue by learning image models with built-in invariance to certain linear transformations, such as rotations. Furthermore, our approach induces transformation-aware features that allow to derive equivariant feature representations *(a,d)*, *i. e.* it is possible to predict how a transformation of the input transforms the feature activations: The feature response *(a)* for 8 orientations of a learned feature for the image patch marked in red already tells us the transformed feature response *(d)* when the input is rotated *(c)*.

equivariant[1] feature representation (see Fig. 5.1(a,d) for an illustration). Feature learning techniques for recognition, on the other hand, have mainly focused on addressing translation in-/equivariance by using convolutional learning architectures [Norouzi et al., 2009; Lee et al., 2009], or on local rotation invariance [Kavukcuoglu et al., 2009].

Similarly, it is desirable that an image restoration algorithm is equivariant to certain input transformations: If the input image was shifted or rotated, one would expect that the restored image is shifted or rotated the same way, but otherwise unchanged. Yet while traditional regularizers, such as total variation, are rotation invariant leading to equivariant denoising, image models based on learned features are typically not (see Fig. 5.1(b,c)).

Here we aim to address invariance and equivariance to *linear image transformations beyond translation*. Although not limited to this setting, we particularly focus on *rotations*, since for many applications this is the most important transformation in-/equivariance beyond translation. We first propose a general framework for incorporating transformation invariances into product models for feature learning. We then demonstrate its application by extending Field of Experts (FoE) image priors [Roth and Black, 2009; Schmidt et al., 2010] to *R-FoEs*, which are invariant to 90° rotations (or multiples thereof) in addition to being translation invariant. Moreover, we show how the methodology can be used to extend convolutional Restricted Boltzmann Machines

---

[1] Formally, a function f is equivariant to a class of transformations $\mathcal{T}$, if for all transformations $\mathbf{T} \in \mathcal{T}$ of the input $\mathbf{x}$, we can predict a corresponding transformation $\mathbf{T}'$ of its output, *i. e.* $f(\mathbf{Tx}) = \mathbf{T}'f(\mathbf{x})$. Moreover, f is invariant to transformations $\mathcal{T}$ if $f(\mathbf{Tx}) = f(\mathbf{x})$ for all $\mathbf{T} \in \mathcal{T}$.

(C-RBMs) [Norouzi et al., 2009; Lee et al., 2009] to *RC-RBMs*, which are translation and rotation invariant.

While invariances can be learned directly from training data, this may require inordinate amounts of data. But even if the training data was sufficient to learn invariances without any model provisions, then some of the learned features would be transformed versions of others to account for this invariance [Welling et al., 2003]. One important shortcoming of this approach is that it is unclear how the different features are related in terms of the image transformations between them. This makes it difficult to build in-/equivariant feature *descriptors* for invariant object recognition or detection from them. A key property of our approach is that it allows to induce *transformation-aware* features, *i. e.* we can predict how the feature activations change as the input image is being transformed, which we further exploit to define a *rotation-equivariant feature descriptor*, called *EHOF*, based on features learned with an RC-RBM. We also extend EHOF to a fully *rotation-invariant descriptor*, *IHOF*.

A descriptor *is a representation of an image (region) as typically used for classification or retrieval.*

We demonstrate the benefits of our approach in two applications. First, we show how learning a rotation-invariant image prior benefits equivariant image restoration. Second, we apply the learned features as well as the proposed rotation in-/equivariant descriptors in the context of object recognition and detection. We test our approach on two challenging data sets for rotation-invariant classification and detection, and in each case outperform state-of-the-art methods from the recent literature.

5.1 PRODUCT MODELS & LINEAR TRANSFORMATIONS

Many probabilistic models of images and other dense scene representations, such as depth and motion, can be seen as product models in which each factor models a specific property of the data that is extracted using a linear feature transform. If we denote the vectorized image as $\mathbf{x} \in \mathbb{R}^n$ and $\mathcal{F} = \{\mathbf{F}_{(i)} \in \mathbb{R}^{m_i \times n} | i = 1, \ldots\}$ as a set of linear feature transformations, we can write an abstract product model as

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{i=1}^{|\mathcal{F}|} \varphi_i\big(\mathbf{F}_{(i)}\mathbf{x}; \theta_i\big). \tag{5.1}$$

Here, the φ_i are the individual factors (potentials) that model the result of each linear feature transform $\mathbf{F}_{(i)}$ based on parameters θ_i, and Z is a normalization constant making $p(\mathbf{x})$ a proper density (from now on omitted for brevity).

Markov random fields (MRFs) can be interpreted as one instance of such a product model (*cf.* Section 2.2) by defining "cropping" matrices: $\mathbf{C}_{(i)}$ crops out a single pixel i from $\mathbf{x}$ such that $\mathbf{C}_{(i)}\mathbf{x} = x_i$, and

$\mathbf{C}_{(k,l)}$ crops out two neighboring pixels k and l such that $\mathbf{C}_{(k,l)}\mathbf{x} = (x_k, x_l)^\mathsf{T}$. Then

$$p_{\mathrm{MRF}}(\mathbf{x}) \propto \prod_{i=1}^{n} \varphi_i\big(\mathbf{C}_{(i)}\mathbf{x}; \theta_i\big) \prod_{(k,l)\in\mathcal{E}} \varphi_{kl}\big(\mathbf{C}_{(k,l)}\mathbf{x}; \theta_{kl}\big) \tag{5.2}$$

denotes a standard pairwise MRF, where φ_i are the unaries, and φ_{kl} the pairwise terms for each edge $(k,l) \in \mathcal{E}$.

If the feature transformation matrices $\mathbf{F}_{(i)}$ are filters (row vectors), *i.e.* $\mathbf{F}_{(i)} = \mathbf{J}_i^\mathsf{T} \in \mathbb{R}^{1\times n}$, that project into a 1D subspace, then we also notice that Eq. (5.1) is a Product of Experts (PoE) with linear feature transforms [Hinton, 2002]:

$$p_{\mathrm{PoE}}(\mathbf{x}) \propto \prod_{i=1}^{|\mathcal{F}|} \varphi_i\big(\mathbf{J}_i^\mathsf{T}\mathbf{x}; \theta_i\big). \tag{5.3}$$

We note that such PoEs with linear experts directly generalize *principal component analysis* (PCA) [*cf.* Jolliffe, 2002], *independent component analysis* (ICA) [*cf.* Hyvärinen and Oja, 2000], as well as *Restricted Boltzmann Machines* (RBMs) [Hinton, 2002] (see also Section 5.3).

Despite the notational similarity, there are two key differences between the pairwise MRF in Eq. (5.2) and linear PoEs or RBMs as in Eq. (5.3). Pairwise MRFs have fixed feature transformations, whereas they are learned from data in case of linear PoEs and RBMs. Moreover, the primary goal of MRFs is usually modeling the prior distribution $p(\mathbf{x})$ itself, *e.g.*, for regularization, but linear PoE models and RBMs often use the probabilistic model only as a tool for learning the features $\mathbf{F}_{(i)}$ for use in other tasks such as recognition.

5.1.1 *Integrating transformation invariance*

To see how product models can be made transformation invariant, it is useful to study the MRF model from Eq. (5.2) in more detail. MRFs in vision are typically made translation invariant by ensuring that the unary terms and the pairwise terms are the same everywhere in the image (*i.e.* φ_i and θ_i do not depend on i, and φ_{kl} and θ_{kl} only depend on the relative position of pixels k and l). In other words, translation invariance is achieved by taking a product of the same unary and pairwise terms over all possible pixel locations. High-order MRFs [Zhu and Mumford, 1997; Roth and Black, 2009] and convolutional RBMs [Norouzi et al., 2009; Lee et al., 2009] do so analogously (*cf.* Sections 5.2 and 5.3).

We here generalize this concept to *arbitrary linear image transformations*. Given a finite set of linear image transformations $\mathcal{T} = \{\mathbf{T}_{(j)} | j = 1, \ldots\}$ of one or more types, we define a transformation-invariant product model w.r.t. $\mathcal{T}$ as

$$p_{\mathcal{T}}(\mathbf{x}) \propto \prod_{j=1}^{|\mathcal{T}|} \prod_{i=1}^{|\mathcal{F}|} \varphi_i\big(\mathbf{F}_{(i)}\mathbf{T}_{(j)}\mathbf{x}; \theta_i\big). \tag{5.4}$$

To achieve invariance, it is important that both the factor φ_i and its parameters θ_i do not depend on $\mathbf{T}_{(j)}$. However, due to the necessarily finite representation of images and the finite transformation class $\mathcal{T}$, such invariances in most cases only hold approximately.

While Eq. (5.4) may seem like an innocuous change over Eq. (5.1), it has several important properties: *(1)* the framework generalizes a known mechanism for translation invariance [Zhu and Mumford, 1997; Norouzi et al., 2009] to arbitrary finite sets of linear transformations $\mathcal{T}$, including rotations; *(2)* unlike other attempts to achieve simultaneous invariance to several transformations, *e. g.*, translation and rotation [Kivinen and Williams, 2011], we treat all transformations equally, and do not introduce additional latent variables [Frey and Jojic, 1999]; *(3)* the formulation is a special case of the generic product model in Eq. (5.1), in which the factors model the responses to the compound linear transformation $\mathbf{F}_{(i)}\mathbf{T}_{(j)}$, and the type and parameters of the factors are shared between all possible transformations in $\mathcal{T}$; *(4)* transformation invariance can be added to a wide range of product models without substantial modifications to their algorithmic backbone for learning and inference; *(5)* since the factors and their parameters are shared between all transformations, this leads to parsimonious representations with comparatively few parameters that may also be easier to interpret; and finally, *(6)* this will later allow us to construct equivariant descriptors with learned features, which in turn facilitate rotation-invariant object detection.

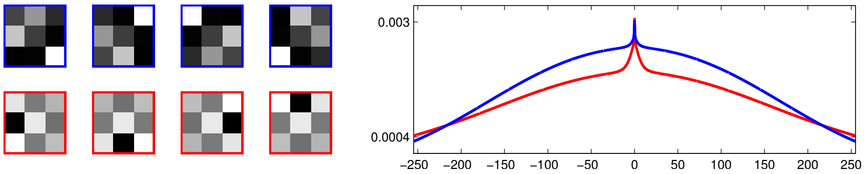
5.2 LEARNING ROTATION-INVARIANT IMAGE PRIORS

Many problems in low-level vision require prior knowledge. In image restoration tasks, such as denoising, deblurring, or inpainting, image priors are crucial for recovering a plausible image from noisy, blurred, or incomplete inputs. While traditionally pairwise MRFs (Eq. 5.2) have been the prevalent probabilistic prior model of images [Li, 2001], recent years have seen an increased adoption of learned high-order priors [Zhu and Mumford, 1997; Roth and Black, 2009]. They not only benefit from modeling complex image structure in large patches (cliques), but also from learning the model parameters from training data.

It is important to note that several popular image priors can be seen as special cases of our transformation-invariant learning framework. To that end we define a set of "convolutional" transformations as

$$\mathcal{T}_C = \left\{ \mathbf{C} \cdot \mathbf{S}_{(k,l)} \middle| k = 1, \ldots, r,\ l = 1, \ldots, c \right\}, \tag{5.5}$$

where the linear transformation $\mathbf{S}_{(k,l)}$ translates the image such that pixel (k, l) is at the origin, and $\mathbf{C}$ crops a fixed size image patch (*e. g.*, 3×3 pixels) around the origin. Here, $\mathbf{S}_{(k,l)}$ achieves translation in-



(a) 2 features $\mathbf{J}_i \times 4$ rotations (b) 2 factors (experts) φ_i

Figure 5.2: **Learned R-FoE model with 2 experts and 4 rotations.** The features and corresponding expert shapes are color-matched.

variance, while $\mathbf{C}$ ensures that the model complexity is independent of the image size.

It is now quite straightforward to see that the FRAME model [Zhu and Mumford, 1997] and the FoE [Roth and Black, 2009] are special cases of Eq. (5.4) with $\mathcal{T} = \mathcal{T}_\mathrm{C}$. In FRAME, the feature transformations $\mathbf{F}_{(i)}$ are hand-chosen filters and the factors φ_i are learned from data. The FoE additionally learns the linear features $\mathbf{F}_{(i)} = \mathbf{J}_i^\mathsf{T}$ from data.

However, the FoE is not explicitly designed to incorporate any invariances beyond image translations. Since the features are unconstrained during learning, it is for example not guaranteed that horizontal and vertical image structure is modeled equally, which can be argued is a desirable property of an image prior: The quality of a restored image should be the same, regardless of whether the image was restored in portrait or landscape orientation. As Fig. 5.1 shows, rotating an image by 90° may already substantially change the energy of the image under the non-invariant prior.

We propose to additionally impose the desired invariance to rotations into the model, and define the transformation set as

$$\mathcal{T}_\mathrm{RC} = \left\{ \mathbf{R}_{(\omega)} \cdot \mathbf{C} \cdot \mathbf{S}_{(k,l)} \,\middle|\, {}^{\omega \in \Omega,}_{k=1,\ldots,r,\; l=1,\ldots,c} \right\}. \tag{5.6}$$

Here $\mathbf{R}_{(\omega)}$ performs an image rotation of the cropped patch by angle ω, and $\mathbf{S}_{(k,l)}$ and $\mathbf{C}$ are defined as before. Using the transformation set $\mathcal{T}_\mathrm{RC}$ – here with 90° rotation increments, *i. e.* $\Omega = \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$ – we train a rotation-invariant FoE image prior (*R-FoE*)

$$p_\text{R-FoE}(\mathbf{x}) \propto \prod_{\omega \in \Omega} \prod_{(k,l)} \prod_{i=1}^{|\mathcal{F}|} \varphi_i\left(\mathbf{J}_i^\mathsf{T} \cdot \mathbf{R}_{(\omega)} \cdot \mathbf{C} \cdot \mathbf{S}_{(k,l)} \mathbf{x}; \theta_i\right) \tag{5.7}$$

with $|\mathcal{F}| = 2$ features (filters) $\mathbf{J}_i$ defined on 3×3 patches. The factors (experts) φ_i are modeled as Gaussian scale mixtures (as in Chapter 4), and learning is done using contrastive divergence and Gibbs sampling (*cf.* Chapter 2), exploiting the half-quadratic representation of the R-FoE model. Fig. 5.2 shows the 2 learned features with their 4 implicitly induced rotations (as an effect of the $\mathbf{R}_{(\omega)}$), and the corresponding experts. Note that the 4 different rotations share the same expert (and parameters), which ensures that the learned model is fully invariant to image rotations in 90° increments. While finer-grained invariance with smaller angular increments is in principle possible, this necessitates larger filters, which remains challenging due to filters and experts being learned simultaneously.

5.3 LEARNING ROTATION-AWARE IMAGE FEATURES

Besides transformation-invariant image models, our second main goal is to learn transformation-aware image features that will later allow us to derive transformation in-/equivariant feature descriptors for object detection[2]. A widely used model for feature learning is the Restricted Boltzmann Machine (RBM) [Hinton, 2002]. For a binary image $\mathbf{x}$ and a set of binary hidden variables $\mathbf{h} \in \{0,1\}^K$ it is defined as

$$p_{\text{RBM}}(\mathbf{x}, \mathbf{h}) \propto \exp\left(\mathbf{c}^\mathsf{T}\mathbf{x}\right) \prod_{i=1}^{K} \exp\left(h_i\left(\mathbf{w}_i^\mathsf{T}\mathbf{x} + b_i\right)\right). \tag{5.8}$$

If the image $\mathbf{x}$ is real-valued, a Gaussian RBM is typically used instead and defined as

$$p_{\text{GRBM}}(\mathbf{x}, \mathbf{h}) \propto \exp(-\|\mathbf{x}\|^2/2) \cdot p_{\text{RBM}}(\mathbf{x}, \mathbf{h}), \tag{5.9}$$

after the visible units (*i.e.*, pixels) are scaled to unit variance in a preprocessing step. Note that the Gaussian RBM admits HQ inference (of the integral type) in the sense that $p_{\text{GRBM}}(\mathbf{x}|\mathbf{h}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_\mathbf{h}, \mathbf{I})$ is a Gaussian distribution with identity covariance matrix and mean $\boldsymbol{\mu}_\mathbf{h}$ determined by the hidden variables $\mathbf{h}$. However, it does not correspond to one of the HQ forms that we discussed in Chapter 3.

By marginalizing out the hidden variables $\mathbf{h}$ it is possible to rewrite an RBM as a generic product model as in Eq. (5.1):

$$p_{\text{RBM}}(\mathbf{x}) = \int p_{\text{RBM}}(\mathbf{x}, \mathbf{h})\, d\mathbf{h} \propto \exp\left(\mathbf{c}^\mathsf{T}\mathbf{x}\right) \prod_{i=1}^{|\mathcal{F}|} \varphi_i\left(\mathbf{F}_{(i)}\mathbf{x}; b_i\right), \tag{5.10}$$

where the feature transformations $\mathbf{F}_{(i)} = \mathbf{w}_i^\mathsf{T} \in \mathbb{R}^{1 \times n}$ are single image features (filters) written as a row vector, and

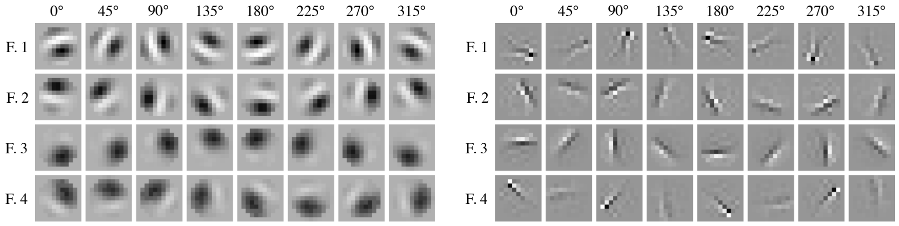
$$\varphi_i(y; b_i) = 1 + \exp(y + b_i) \tag{5.11}$$

is a logistic function with biases b_i of the hidden variables. We keep the biases $\mathbf{c}$ of the visible variables separate and do not make them part of the feature transform.

Standard RBMs are not transformation invariant, but aim to learn pertinent invariances out of the training data, which requires large amounts of data. Moreover, the learned features are not transformation-aware, *i.e.* it is unclear if and how different features relate

---

[2] While it is conceivable to also use the model from Section 5.2 for feature learning itself, the feature size limitations make that currently not practical.



(a) MNIST handwritten digits (b) Natural images (whitened)

Figure 5.3: **Translation- and rotation-aware** 11×11 **image features.** Each row shows one of 4 features, each column one of 8 implicitly induced feature rotations.

in terms of image transformations, which makes it difficult to build in-/equivariant feature descriptors from them. Our goal here is to learn transformation-aware features. The most straightforward invariance/awareness to integrate is w.r.t. image translations. For this we apply our framework from Section 5.1.1 with $\mathcal{T} = \mathcal{T}_C$ (see Eq. 5.5) to the RBM as given in Eq. (5.10) and obtain the known convolutional RBM (C-RBM), which has recently been introduced by several authors [Lee et al., 2009; Norouzi et al., 2009]. C-RBMs naturally extend RBMs to arbitrarily-sized images.

Our contribution is now to generalize C-RBMs to be also invariant to image rotations, which in turn allows to learn features that are both translation- and rotation-aware. To that end we apply our framework to the basic RBM, and use the transformation set $\mathcal{T} = \mathcal{T}_{RC}$ from Eq. (5.6):

$$p_{\text{RC-RBM}}(\mathbf{x}) \propto \exp\left(\mathbf{c}^\mathsf{T}\mathbf{x}\right) \prod_{\omega \in \Omega} \prod_{(k,l)} \prod_{i=1}^{|\mathcal{F}|} \varphi_i\left(\mathbf{w}_i^\mathsf{T} \cdot \mathbf{R}_{(\omega)} \cdot \mathbf{C} \cdot \mathbf{S}_{(k,l)}\mathbf{x}; b_i\right). \tag{5.12}$$

This RC-RBM can also be generalized to continuous-valued images following Eq. (5.9), hence admits HQ inference. Note that the bias terms b_i are shared across all image locations and orientations. If the biases $\mathbf{c}$ of the visible variables differ across the image, Eq. (5.12) will not be invariant to global image rotations. This is not an issue if the goal is to extract locally equivariant features. If global invariance is desired, we can define $\mathbf{c} = c \cdot \mathbf{1}$.

Note that the R-FoE from Eq. (5.7) and the RC-RBM from Eq. (5.12) are actually very similar. The major difference are the factors φ_i, that in case of the R-FoE penalize large filter responses, whereas the RC-RBM encourages large filter responses.

To train the RC-RBM model, we adapt the contrastive divergence-based learning algorithm for C-RBMs of Norouzi et al. [2009]. No tiling is used; each visible unit corresponds to one pixel. In the examples shown in Fig. 5.3, we learn 4 features of 11×11 pixels on MNIST binary handwritten digit images [LeCun and Cortes] (a) and on whitened natural images (b). We use 8 equidistant rotation angles $\Omega = \{0°, 45°, 90°, \ldots, 315°\}$ for a more fine-grained rotation invariance. The matrices $\mathbf{R}_{(\omega)}$ rotate each image patch using bilinear interpolation. To avoid interpolation artifacts in the corners, we only define the feature inside a circular area (visible in Fig. 5.3(a)).

Whitening *refers to decorrelation of the image pixels [*cf. *Hyvärinen et al., 2009, § 5].*

The RC-RBM has several advantages: It yields transformation-aware features, which allow to predict how the feature activations change when the input is shifted or rotated. It also encourages separate features not to be translations and rotations of one another, since these are already implicitly induced. In this way it leads to a parsimonious and statistically efficient representation [*cf.* Bergstra et al., 2011]. Note that feature extraction with RC-RBMs also does not lead to a higher computational cost than with C-RBMs, since a comparable number of effective features are used in practice.

OTHER RELATED WORK Kivinen and Williams [2011] generalize C-RBMs toward rotation-equivariant feature learning, but treat translations and rotations differently – translations in a product framework and rotations using a mixture model. In contrast, our approach is generic and treats all transformations consistently, which for example allows us to rely on existing learning and inference algorithms. Moreover, we apply our method to rotation-equivariant image restoration and object detection. Concurrent to our publication of [Schmidt and Roth, 2012] and related to our approach, Sohn and Lee [2012] proposed a transformation-invariant RBM w.r.t. linear transformations. However, they enforce that only one hidden unit is active among the transformations of a particular feature. Welling et al. [2003] and Kavukcuoglu et al. [2009] learn topographic representations, which allow to assess when two features correspond to similar transformations (*e.g.*, similar rotation angles). By combining feature learning with pooling functions [Kavukcuoglu et al., 2009], one can obtain locally invariant features. It is not straightforward to extend this to global transformation-equivariance, as is achieved here. With a focus on theoretical understanding, Lenc and Vedaldi [2015] recently studied popular image representations w.r.t. equivariance, invariance, and equivalence.

5.4 ROTATION IN-/EQUIVARIANT IMAGE DESCRIPTOR

A simple approach for rotation-invariant object recognition or detection is to model the object class at a canonical orientation and then search over all possible orientations of/in the given image. In practice this is generally not feasible, since at least a traditional feature descriptor would have to be computed at every rotation that is being searched over. At the other end of the spectrum are rotation-invariant image features, which avoid costly computation at many orientations.

Unfortunately, these features are usually less powerful at describing the image content, since the class of features that can be considered is restricted. A trivial example is simply using the image intensities or color values. Another approach are annular histogram bins defined by the area between two concentric circles, which allow for rotation-invariant spatial pooling of image features, a strategy for example used by RIFT [Lazebnik et al., 2004], but known to limit expressiveness [Takacs et al., 2010].

EQUIVARIANT FEATURES A tradeoff between the two extremes is offered by rotation-equivariant image features, where a rotation of the input image results in a predictable transformation of the feature activation, which can usually be carried out with little computational effort (*e.g.*, circular shift operations, see Fig. 5.1). Hence, a rotation-invariant comparison between two image descriptors can be performed quite efficiently (*e.g.*, used by RIFF-Polar [Takacs et al., 2010]).

Standard image gradient features, as used by many image descriptors [*e.g.*, Lowe, 2004; Dalal and Triggs, 2005], have this desirable rotation-equivariance property, which is often exploited. One can, for example, describe the orientations of gradients relative to the dominant orientation at the center of the image patch, thus making the descriptor rotation invariant (*e.g.*, used by RIFT [Lazebnik et al., 2004]). However, this relies on the assumption that there is a dominant gradient orientation at the patch center, which is true for interest points, but not necessarily for dense feature computation, which is common in sliding-window object detection.

Conventional learned features are difficult to use in this way, since it is not known if and which learned features are rotations of each other, and thus difficult to predict the feature activations given a particular rotation of the image. We now describe a powerful rotation-equivariant descriptor that leverages our rotation-aware RC-RBM features.

Note that additional details are available in Appendix A.1.4.

EQUIVARIANT DESCRIPTOR (EHOF) After extracting features using the RC-RBM from Section 5.3 densely at all locations and orientations (45° increments), we perform non-maximum suppression (NMS) over all orientations for each feature and location. This is akin to standard oriented gradient computation (*e.g.*, in HOG [Dalal and Triggs, 2005]) and significantly increases performance. We then spatially pool (histogram) the NMS results on a polar grid covering the whole image or bounding box, with the intention of converting image rotations to spatial translations of the descriptor. Similar to Takacs et al. [2010], we use equidistant cell centers (in angle and radius) in polar coordinates (Fig. 5.4, top left); please note that we allow for an arbitrary number of rings R, cells C, and feature orientations O.

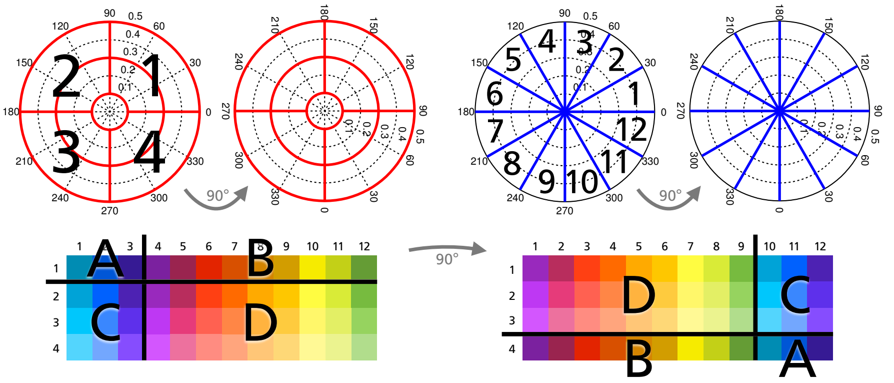


Figure 5.4: **Simplified descriptor example.** The spatial polar grid (red, left) is divided into $R = 2$ rings with $C = 4$ cells each, besides the central cell, which is treated differently (see text); local image features are computed at $O = 12$ orientations (blue, right). After feature extraction and spatial pooling, the histogram values from all rings can be arranged in a single table (bottom, only one ring shown). The rotation of the image and thus the polar grid (here 90°) results in a cyclical 2D translation of the values in the table, as indicated by the colors and regions denoted A–D.

The orientation histogram bins in each cell correspond to the rotation angles of the image features; it is important to arrange the histogram bins in order and with equidistant rotation angles apart (Fig. 5.4, top right).

We then unroll the 3-dimensional histogram $\mathbf{H}_3 \in \mathbb{R}^{R \times C \times O}$ (2 spatial and 1 feature orientation dimension) into the 2-dimensional histogram $\mathbf{H}_2 \in \mathbb{R}^{R \cdot C \times O}$: All spatial cells are assigned a unique ordering by arranging cells from different rings but with neighboring radii together in the rows of the feature matrix $\mathbf{H}_2$ (*i.e.*, first cell 1 from all rings, then all 2nd cells, *etc.*). The columns of $\mathbf{H}_2$ correspond to the histogram orientation bins.

This descriptor layout now has the desired property that a rotation of the image corresponds to a 2-dimensional cyclical translation of the matrix contents (Fig. 5.4, bottom). If the image is rotated by a multiple of the angular distance between neighboring cell centers in the polar grid, this property holds exactly, and approximately in case of all other rotations. To reduce aliasing artifacts in case of rotations that are not aligned with the polar grid, we use bilinear interpolation in polar coordinates for the spatial pooling. Also, the number of cells per ring should be a multiple of the number of histogram orientation bins (or the other way around), otherwise the translations of rows and columns do not match. An important property of this construction is that a rotation of the input image – and thus translation of the matrix $\mathbf{H}_2$ – does not destroy the relative distribution of spatial locations *and* different orientations. Note that the central cell is a special case, since it does not change its spatial location when the input is rotated;

only its histogram orientation bins undergo a 1-dimensional cyclical translation.

We term this descriptor an *equivariant histogram of oriented features* (*EHOF*) to emphasize that it can be built from any locally rotation-equivariant feature, including image gradients and steerable filters [Freeman and Adelson, 1991], to yield a globally rotation-equivariant representation.[3]

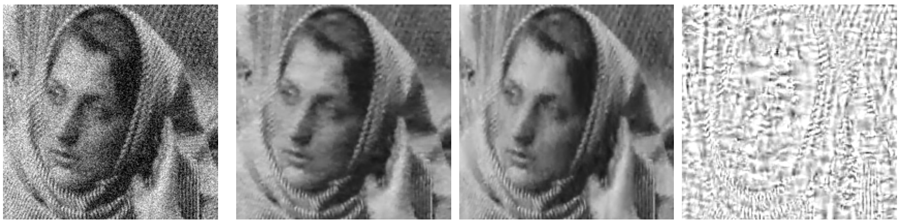
INVARIANT DESCRIPTOR (IHOF) To perform rotation-invariant recognition or detection with this rotation-equivariant descriptor, we could compare two descriptors by defining a custom distance metric as the minimum over all cyclical, 2-dimensional translations between two descriptors (where one of the two is held fixed) that are consistent with an image rotation. A similar strategy is pursued by Takacs et al. [2010], but since rotation-invariant features are used there, the search reduces to 1-dimensional cyclical shifts of their descriptor vector. An obvious disadvantage is the computational cost for this search (for EHOF over several cyclical, 2-dimensional shifts of the feature matrix). Another issue of embedding rotation invariance in the distance computation is that classification algorithms need to be adapted to this case. A preferable solution is thus to make the descriptor itself invariant. To that end, we compute the 2-dimensional discrete Fourier transform (DFT) of the descriptor matrix and only retain its magnitude, which is well-known to be invariant to cyclical shifts; the same can be done in 1D for the central cell. We term the resulting descriptor an *invariant histogram of oriented features* (*IHOF*). Exploiting the translation invariance of the DFT magnitude has the desired advantage of reducing the computational effort, since it only has to be computed once. Moreover, the IHOF descriptor can be directly used in existing classification frameworks.[3]

While the IHOF descriptor is invariant to rotated inputs, we note that it also remains unchanged for other input transformations, which are presumably unlikely for real images as our experimental findings indicate (Section 5.5).

OTHER RELATED WORK Using the magnitude of the 1-dimensional DFT to build rotation-invariant descriptors is used by Ahonen et al. [2009] for local binary pattern histograms with applications to classification and recognition. Employing a log-polar transform to convert rotation and scale changes of an image patch to 2D descriptor translations is commonplace in image registration [*cf.* Zokai and Wolberg, 2005]; this includes using the 2D-DFT to retain invariance to rotation and scale variations. Kokkinos and Yuille [2008] use the 2D-DFT of the log-polar transform to obtain rotation and scale-invariant image descriptors. One difference of such previous approaches to ours is that

---

[3] MATLAB code is available on the author's webpage.

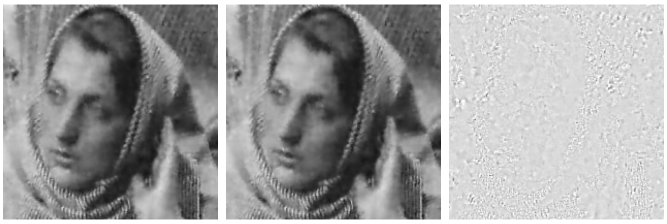


(a) Noisy, 18.67dB (σ=30)

(b) FoE, 25.81dB

(c) FoE, 26.18dB (90° rot.)

(d) $\langle \Delta(\text{b,c})^2 \rangle$=11.63



(e) R-FoE, 26.19dB

(f) R-FoE, 26.19dB (90° rot.)

(g) $\langle \Delta(\text{e,f})^2 \rangle$=1.24

Figure 5.5: **Denoising example (cropped).** *(b,e)* show the results of denoising *(a)*. The results in *(c,f)* are obtained by rotating *(a)* by 90°, denoising the rotated version, and rotating the result back. The results of the non-invariant FoE [Schmidt et al., 2010] in *(b,c)* are sensitive to orientation, both visibly and quantitatively (PSNR difference 0.37dB). The difference between the orientations is shown in *(d)*. The proposed rotation-invariant R-FoE *(e,f)* does not suffer from these problems; any difference in *(g)* is due to sampling-based approximate inference. *Best viewed on screen.*

they work around sparse (interest) points in the image, where the log-polar region only describes the local structure. In contrast, we obtain a globally rotation-invariant image descriptor with fine-grained spatial binning. We use the 2D-DFT to achieve simultaneous invariance to changes of the spatial and feature dimensions, caused by an in-plane rotation of the whole image. Based on a 2D Fourier representation of the gradient histogram, Liu et al. [2014] more recently extend the popular HOG descriptor [Dalal and Triggs, 2005] to be rotation-invariant; using spherical harmonics, they further extend this to 3D invariance.

5.5 EXPERIMENTS

We show the benefits of our framework for *(1)* learning rotation-invariant image priors, and *(2)* for learning equivariant features for recognition and detection, both with and without explicit rotation invariance.

See Appendix A.1.5 for additional experimental details.

INVARIANT IMAGE DENOISING In order to demonstrate the advantage of building explicit invariance to (multiples of) 90° image rotations into learned image priors, we denoise 10 images (from Schmidt et al. [2010]) both in their original orientation, as well as after rotating them by 90°. We compare the FoE implementation of Schmidt et al. [2010] (8 unconstrained features with 3×3 pixels), which does not explicitly enforce rotation invariance, to the R-FoE model proposed in Section 5.2 (8 effective features obtained from 2 learned filters with 3×3 pixels in 4 rotations); denoising is performed using sampling-based MMSE estimation in both cases (*cf.* Section 4.3).

We find that the average performance (PSNR) of an FoE without built-in rotation invariance deteriorates on the rotated images from 32.88dB to 32.77dB (σ=10) and from 28.91dB to 28.75dB (σ=20). In contrast, our rotation-invariant R-FoE achieves exactly the same denoising results of 32.80dB (σ=10) and 28.89dB (σ=20) on original and rotated images, as expected. Furthermore, both models achieve comparable results on non-rotated images, despite the R-FoE having only $\frac{2}{8}$ as many parameters. Fig. 5.5 visualizes the difference between both models.

HANDWRITTEN DIGIT RECOGNITION To establish a performance baseline for the rotation-aware features learned using the proposed RC-RBM, as well as for the rotation in-/equivariant descriptors, we compare against other feature learning approaches, and also use oriented image derivatives ("gradients") with our descriptors. We always use our descriptor with 1 ring and 8 cells (plus central cell) and extract features at 8 orientation angles. The corresponding EHOF descriptors for each of the 4 learned features (Fig. 5.3(a)) have 72 dimensions, which we concatenate to represent each digit. We train the RC-RBM on the MNIST handwritten digit dataset [LeCun and Cortes], which contains 60000 binary training and 10000 test images, and use an rbf-SVM (Support Vector Machine [Cortes and Vapnik, 1995] with a radial basis function kernel [*cf.* Vert et al., 2004]) for classification. Tab. 5.1 gives the recognition results for our method and various competing approaches from the literature. Despite having a parsimonious representation and only using a single model "layer", our approach (EHOF) is competitive with multilayer feature learning approaches, including deep belief networks; somewhat surprisingly, this even holds for simple image derivatives as the sole image feature (akin to HOG [Dalal and Triggs, 2005]). Combining learned features with gradients results in an additional improvement, showing that different properties of the data are captured by each of them. For reference, we also report results with IHOF for MNIST and observe reduced performance, as expected, since MNIST digits do not appear at arbitrary orientations. Otherwise, we see similar behavior, although the RC-RBM features give much better results in this scenario as compared to gradients.

Model / Features	MNIST	MNIST-rot
Multilayer C-RBM, SVM [Lee et al., 2009]	0.82%	—
Multilayer C-RBM, rbf-SVM [Norouzi et al., 2009]	0.67%	—
Deep belief network [Hinton and Salakhutdinov, 2006]	1.20%	—
Deep belief network (best from [Larochelle et al., 2007])	—	10.30%
SDAIC [Larochelle et al., 2009]	—	8.07%
Sparse TIRBM [Sohn and Lee, 2012]	—	4.20%
EHOF (Gradients)	0.97%	5.20%
EHOF (RC-RBM)	0.85%	6.36%
EHOF (RC-RBM + Gradients)	0.62%	4.75%
IHOF (Gradients)	5.82%	8.13%
IHOF (RC-RBM)	2.66%	5.47%
IHOF (RC-RBM + Gradients)	2.26%	3.98%

Table 5.1: **Test error** on MNIST [LeCun and Cortes] and MNIST-rot [Larochelle et al., 2007].

In order to show the benefits of making the rotation-equivariant EHOF descriptor rotation-invariant by using its DFT magnitude, we evaluate the performance on the MNIST-rot dataset [Larochelle et al., 2007], containing 12000 images for training and validation, and 50000 test images, in which digits appear at all orientations. Tab. 5.1 gives the results (following the protocol of Larochelle et al. [2007]) and compares to state-of-the-art techniques from the literature. Even with the EHOF descriptor, we achieve superior results than most competing approaches since the rbf-SVM is able to learn necessary invariances from the data. Gradients yield better results than RC-RBM features with EHOF, although the situation is reversed when comparing IHOF performance. Either way, in both cases we gain a substantial improvement when combining image gradients with our learned features. It is important to note that the learned features (alone and combined with gradients) always yield superior results with IHOF. Combining the IHOF descriptors computed from RC-RBM features and image derivatives results in a competitive test error of 3.98%, which is about 50% lower than the previous best result [Larochelle et al., 2009] that we were aware of when [Schmidt and Roth, 2012] was published, and comparable to the related approach of Sohn and Lee [2012], which was proposed concurrently to ours.

AERIAL CAR DETECTION Most feature learning approaches from the literature, [Norouzi et al., 2009] being a notable exception, only report results for object classification. In contrast, we demonstrate the use of our RC-RBM features and the IHOF image descriptor for rotation-invariant object detection, specifically for finding cars in satellite imagery. We use the dataset introduced by Heitz and Koller

[2008], which consists of 30 images, containing a total of 1319 cars that occur at arbitrary orientations and are only annotated with axis-aligned bounding boxes. We perform 5-fold cross validation and report average results across all folds.

Based on a simple and efficient linear SVM classifier, we train a sliding-window detector [Dalal and Triggs, 2005] with fixed window size of 40 × 40 pixels. We use an RC-RBM trained on natural images to extract 4 translation- and rotation-aware features (Fig. 5.3(b)), each pooled in the EHOF descriptor on a polar grid with 3 rings and 16 cells per ring (plus central cell), and a histogram over 8 feature orientations for each cell. The combined EHOF descriptors have 1568 dimensions in this case. The rotation-invariant IHOF descriptor is obtained using the 2D-DFT magnitude.

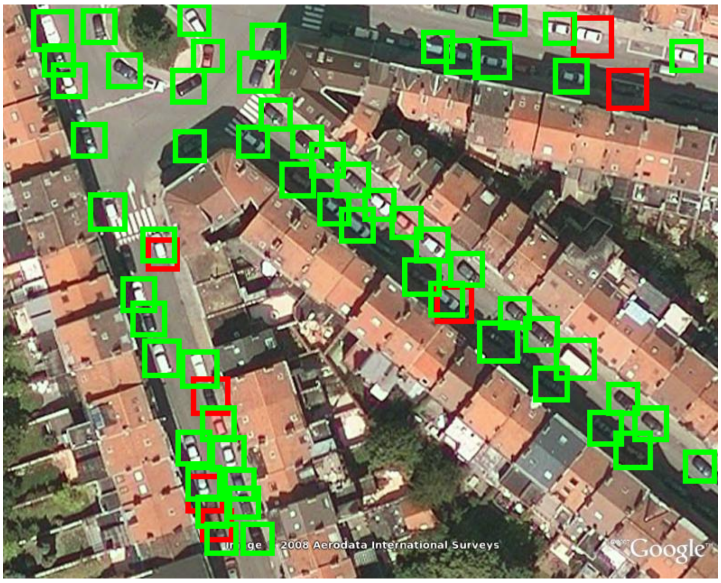
As Fig. 5.6 shows, our IHOF descriptor substantially increases the detection performance over a standard HOG descriptor (also with a linear SVM) from 54.5% average precision (AP) to 72.7%. For reference, we also report the results of using the EHOF descriptor, which underline the benefits of using the rotation-invariant IHOF descriptor for this task. Since the learned RC-RBM features are not as localized as the gradient features used in the successful HOG descriptor, we also evaluated the use of simple gradient features in the rotation-invariant IHOF descriptor. This leads to an improved performance of 74.7% AP, which is close to the recent approach of Vedaldi et al. [2011]. Their approach is much more complex and uses structured output SVM regressors and non-linear kernels to achieve 75.7% AP. Note that we also clearly outperform the context-based approach of Heitz and Koller [2008]. More importantly, the RC-RBM features again contain information that is complementary to gradient features. Combining both boosts the performance to 77.6%, which is a clear improvement over the best performance reported in the literature (75.7% AP [Vedaldi et al., 2011]) at the time when [Schmidt and Roth, 2012] was published. Since then, Liu et al. [2014] obtained improved results of 82.6% (84.2%) AP with a linear SVM (Random Forest [Breiman, 2001]) classifier and their rotation-invariant Fourier HOG features.

Still, we expect to obtain better results with more advanced variants of RBMs [*e.g.*, Courville et al., 2011], or through stacking to obtain deep models. Furthermore, adapting descriptors to features plays an important role for recognition/detection performance, which so far has mostly been explored for gradient features. Hence, an interesting avenue for further research is descriptor learning [*e.g.*, Brown et al., 2011].

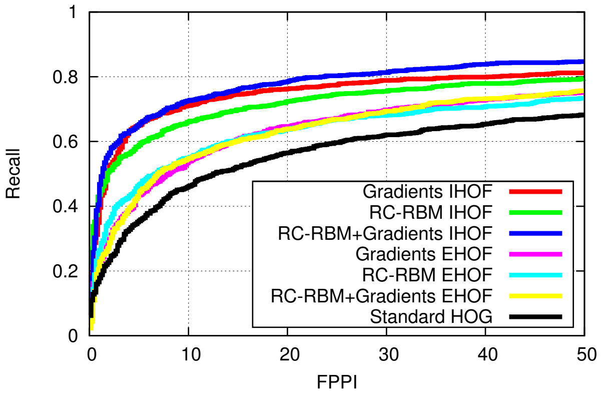
5.6 SUMMARY

We proposed a framework for transformation-invariant feature learning using product models, demonstrated how popular translation-invariant models are special cases, and studied its application to inducing rotation invariance. We extended a learned image prior to be (90°) rotation-invariant, and showed its advantages over a conventional prior. We also applied our framework to make convolutional RBMs rotation-invariant, and used this RC-RBM for translation- and rotation-aware feature learning. In both cases, we exploited half-quadratic representations of the respective product models to make learning and inference practical.

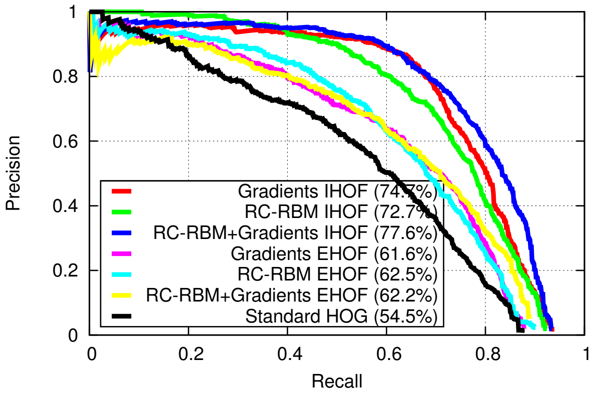
Finally, we employed the learned features, or other oriented features, to build a globally rotation-equivariant image descriptor (EHOF), which can be made rotation-invariant (IHOF) using the 2D-DFT magnitude. We demonstrated state-of-the-art results on two challenging datasets for rotation-invariant recognition and detection.



(a) Detection examples



(b) FPPI vs. Recall



(c) Recall vs. Precision

Figure 5.6: **Aerial car detection.** *(a)* Example of detections with the *IHOF* descriptor encoding *RC-RBM+Gradients* features, where green boxes indicate correct detections and red boxes incorrect ones. *(b,c)* Common performance measures (FPPI stands for *false positives per image*).

6

CASCADES OF GAUSSIAN CONDITIONAL RANDOM FIELDS

CONTENTS

6.1 Generalizing Half-quadratic Regularization **132**
  6.1.1 Discriminative alternative 133
  6.1.2 Discriminative model cascade 135
6.2 Gaussian CRF for Restoration **136**
  6.2.1 Regression tree field 138
  6.2.2 Training 139
  6.2.3 RTF prediction cascade 140
6.3 Experiments **142**
  6.3.1 Image denoising 142
  6.3.2 Image deblurring 146
6.4 Discussion **150**
  6.4.1 Training dataset 150
  6.4.2 Model connectivity and comparison 151
6.5 Summary **153**

Image restoration is an important and long-studied field, manifesting itself in numerous applications, such as image denoising, deblurring, or super-resolution. Recall that image restoration can be seen as an *inverse problem*, where an image corruption process – modeled by a data (or likelihood) term – is to be inverted. Such an inversion is typically mathematically ill-posed, which necessitates the use of regularization (or prior knowledge).

Prior knowledge can be imposed in a variety of ways. Discriminative approaches have received increasing attention in recent years, particularly for image denoising [Tappen et al., 2007; Barbu, 2009; Jancsary et al., 2012a; Burger et al., 2012], where they often yield state-of-the-art restoration performance combined with low computational effort. In other image restoration applications, such as non-blind image deblurring [Levin et al., 2007; Krishnan and Fergus, 2009; Schmidt et al., 2011] on the other hand, generative approaches have been standard. We argue that the lack of discriminative methods for these applications stems from their more challenging data term with additional instance-specific parameters, which are necessary to capture the image corruption process properly. In non-blind deblurring[1],

---

[1] A more precise term would be *deconvolution* instead of deblurring when a stationary blur assumption is made. We use the more general terminology as our discussion is not limited to deconvolution.

for example, not only the noise level, but also the blur kernel has to be given to the algorithm, which is often only known at test time and may vary from (image) instance to instance (*cf.* Chapter 4). For deblurring the instance-specific parameters thus correspond to the blur kernel. In a discriminative approach it is, however, quite difficult to cope with such instance-specific parameters.

In this chapter we introduce a *discriminative* image restoration approach for applications that can be expressed via arbitrary quadratic data terms (Gaussian likelihoods). The first major challenge we address is that the number of possible inputs to such a model increases exponentially with the number of (input) parameters. Because of that, training a conditional model for every possible instance-specific parameter, *e.g.* for every possible blur kernel [Schuler et al., 2013], is very costly or even infeasible. We thus argue that it is important to be able to train a *single* model that outputs a restored image given an arbitrary input image *and* instance parameter, such as the blur kernel. We address the challenge of capturing the input distribution variability by using a semi-parametric approach: We specify part of the model explicitly by means of instance-specific parameters and capture the remaining variability using non-parametric regression trees. As a consequence, we assume access to these instance-specific parameters during training and testing, for example by means of an estimate. More specifically, our approach is based on regression tree fields (RTFs) [Jancsary et al., 2012b], a Gaussian conditional random field (CRF) in which the parameters of the Gaussian field are determined through regression trees. However, we could alternatively make use of other Gaussian CRFs [*e.g.*, Tappen et al., 2007].

When considering image deblurring in contrast to image denoising, a second major challenge arises: The great variability of the image corruption due to blur that is only known at test time makes it also rather difficult to derive suitable features from the input image, which are then used as inputs for the regression trees. To address this we take inspiration from common half-quadratic approaches to image restoration (Chapter 3). In particular, we observe that while half-quadratic MAP estimation makes its final prediction also based on a Gaussian random field, the parameters of this random field are iteratively refined during the inference procedure. This is in contrast to typical Gaussian CRF approaches, where the parameters are estimated in a one-shot fashion. Motivated by that, we introduce a model cascade based on regression tree fields. The first stage predicts a relatively crude estimate that removes dominant image blur, which is however very useful to define better input features for later stages. In this way the deblurred image is incrementally refined in each stage. We apply our discriminative prediction cascade also to the problem of image denoising, where we find that the cascade architecture benefits image quality as well, albeit somewhat less than for deblurring.

Our model cascade is trained discriminatively by minimizing an application-specific loss function (here, PSNR) on a training set (*cf.* Section 2.4). To make this feasible, we synthesize training data according to the given application-specific data term. One challenge in case of deblurring is that sufficient training data must be available for discriminative training, but realistic image blurs are quite scarce [Levin et al., 2009; Köhler et al., 2012]. To overcome this limitation, we use synthetically generated blur kernels based on a simple motion model, which we show to generalize well to kernels encountered in practice.

CONTRIBUTIONS We make the following contributions: *(1)* We analyze commonly used half-quadratic regularization [Geman and Yang, 1995; Geman and Reynolds, 1992] with sparse image priors, and draw connections to discriminative prediction with a CRF; *(2)* we introduce a discriminative prediction cascade for image restoration based on regression tree fields, which naturally arises as a generalization of half-quadratic inference; *(3)* we employ a semi-parametric approach at each prediction stage, which allows a single trained model to cope with parameters that vary from instance to instance, such as the blur kernel in image deblurring; *(4)* we train our model with data that was obtained by using realistic, but synthetically generated blur kernels and experimentally demonstrate that the trained model generalizes to unseen real blur kernels; *(5)* we demonstrate state-of-the-art performance on a synthetically blurred test set [Schmidt et al., 2011] and on two realistic data sets for camera shake [Köhler et al., 2012; Levin et al., 2011]. While previous non-blind deblurring approaches have for the most part either been very fast but with inferior performance, or slow but with high-quality results [*e. g.*, Schmidt et al., 2011], our approach delivers state-of-the-art deblurring performance with an efficient inference method that allows deblurring images of moderate resolution in a reasonable amount of time; *(6)* we demonstrate state-of-the-art performance for a (grayscale) image denoising benchmark. We also train our model for color denoising and show its superiority over applying our grayscale denoising model independently to each color channel.

NON-BLIND DEBLURRING To this date, most approaches rely on the classical Lucy-Richardson algorithm as non-blind deblurring component [*e. g.*, Fergus et al., 2006], or use manually-defined MRF image priors [Levin et al., 2007; Krishnan and Fergus, 2009; Xu and Jia, 2010]. Generatively-trained MRF priors applied to non-deblurring [Schmidt et al., 2011] have found limited adoption due to computational challenges from inference. In this chapter we assume stationary image blur, *i. e.* the observed image is the result of convolving the unknown original image with a blur kernel (+ noise), but our approach is not limited to this setup and can be extended to non-uniform image blurs.

To motivate our discriminative approach and understand its connections to the existing literature, it is beneficial to recall half-quadratic (HQ) inference (Chapter 3) and its relation to recent image restoration approaches. In image deblurring, denoising and other restoration applications, sparse image priors are frequently used for regularization [*e.g.*, Levin et al., 2007; Roth and Black, 2009; Krishnan and Fergus, 2009]. Typically, they can be seen as specific instances of the Field of Experts (FoE) prior (Section 2.2.2) and model an image $\mathbf{x}$ through the response of linear filters $\mathbf{f}_j$ (*e.g.*, horizontal and vertical image derivatives), which induce overlapping cliques $c \in \mathcal{C}_j$ in the corresponding MRF prior (*cf.* Eq. (3.2)):

$$p(\mathbf{x}) \propto \prod_j \prod_{c \in \mathcal{C}_j} \exp\big(-\rho_j(\mathbf{f}_j^\mathsf{T}\mathbf{x}_{(c)})\big). \tag{6.1}$$

A sparse (non-Gaussian) potential function $e^{-\rho_j}$ models the filter response of $\mathbf{f}_j$ to the clique pixels $\mathbf{x}_{(c)}$.

As before, we make the typical assumption that the image corruption process is modeled by specifying a Gaussian likelihood $p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2\mathbf{I})$ for the observed, corrupted image $\mathbf{y}$. In the case of non-blind deconvolution, we have $\mathbf{K}\mathbf{x} \equiv \mathbf{k} \otimes \mathbf{x}$, where $\mathbf{K}$ is the blur matrix that corresponds to convolving the image with a blur kernel $\mathbf{k}$. The image noise is assumed to be additive white Gaussian noise with variance σ^2. The problem of image denoising arises as a special case with $\mathbf{K} = \mathbf{I}$ being the identity matrix. If multiplication with $\mathbf{K}$ reduces the spatial resolution of the image, the likelihood models the problem of super-resolution. Using Bayes' theorem, one obtains the posterior distribution over the restored image as $p(\mathbf{x}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}) \cdot p(\mathbf{x})$.

We discussed at length in Chapter 3 that HQ inference aims to make posterior inference easier by first augmenting the image prior with auxiliary/latent variables z_{jc}, such that the prior is retained via an operation $\bigoplus \in \{\max, \sup, \sum, \int\}$ that eliminates the additional variables:

$$p(\mathbf{x}) \propto \prod_j \prod_{c \in \mathcal{C}_j} \bigoplus_{z_{jc}} \exp\big(-\phi_j(\mathbf{f}_j^\mathsf{T}\mathbf{x}_{(c)}, z_{jc})\big). \tag{6.2}$$

Since $\bigoplus$ is distributive w.r.t. the product operation, the augmented image prior is obtained as

$$p(\mathbf{x}, \mathbf{z}) \propto \prod_j \prod_{c \in \mathcal{C}_j} \exp\big(-\phi_j(\mathbf{f}_j^\mathsf{T}\mathbf{x}_{(c)}, z_{jc})\big) \tag{6.3}$$

with $p(\mathbf{x}) \propto \bigoplus_{\mathbf{z}} \; p(\mathbf{x}, \mathbf{z})$. Recall that for a fixed setting of $\mathbf{z}$ the distribution $p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{\mathbf{x}|\mathbf{z}}, \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{z}})$ is jointly Gaussian and further yields a Gaussian posterior when combined with a Gaussian likelihood:

$$\begin{aligned} p(\mathbf{x}|\mathbf{y}, \mathbf{z}) &\propto \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \sigma^2\mathbf{I}) \cdot \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{\mathbf{x}|\mathbf{z}}, \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{z}}) \\ &\propto \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y},\mathbf{z}}, \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y},\mathbf{z}}). \end{aligned} \tag{6.4}$$

MAP estimation (Algs. 3.1 and 3.3) can now be carried out on the augmented posterior $p(\mathbf{x}, \mathbf{z}|\mathbf{y})$ by alternating between maximizing $p(\mathbf{x}|\mathbf{y}, \mathbf{z})$ and using $p(\mathbf{z}|\mathbf{x}, \mathbf{y})$ to update the auxiliary variables; the type of update depends on the choice of the operation $\bigoplus$. Maximizing $p(\mathbf{x}|\mathbf{y}, \mathbf{z})$ w.r.t. $\mathbf{x}$ amounts to computing $\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y},\mathbf{z}}$, which requires solving a sparse system of linear equations based on the precision matrix $\boldsymbol{\Sigma}^{-1}_{\mathbf{x}|\mathbf{y},\mathbf{z}}$ (*cf.* Section 3.5). Updating $\mathbf{z}$ based on $p(\mathbf{z}|\mathbf{y}, \mathbf{x})$ is easy, because it can be done for each scalar variable z_{jc} individually (*e.g.*, with a table lookup), since all z_{jc} are independent:

$$p(\mathbf{z}|\mathbf{x}, \mathbf{y}) \propto \prod_{j} \prod_{c \in \mathcal{C}_j} p(z_{jc}|\mathbf{x}, \mathbf{y}) \tag{6.5}$$

$$p(z_{jc}|\mathbf{x}, \mathbf{y}) \propto \exp\big(-\phi_j(\mathbf{f}_j^{\mathsf{T}} \mathbf{x}_{(c)}, z_{jc})\big). \tag{6.6}$$

By using the fact that a wide variety of robust (sparse) potentials ρ_j can be expressed (or approximated) as the envelope of auxiliary functions (Section 3.3.1), one can re-formulate the majority of sparse image priors in this way. Levin et al. [2007] and Krishnan and Fergus [2009] have employed this principle for efficient image deblurring. In Chapter 4, we have used MRF image priors based on GSMs [Wainwright and Simoncelli, 2000], which give rise to a half-quadratic representation of the integral type (Section 3.3.2) with $\bigoplus = \sum$ (or $\bigoplus = \int$ for infinite GSMs). This has been used by Schmidt et al. [2010] for image denoising and deblurring [Schmidt et al., 2011] with sampling-based inference, which alternates between sampling from $p(\mathbf{x}|\mathbf{y}, \mathbf{z})$ and $p(\mathbf{z}|\mathbf{x}, \mathbf{y})$. Babacan et al. [2012] have exploited half-quadratic representations in the context of blind deconvolution.

6.1.1 *Discriminative alternative*

To see how classical half-quadratic regularization can be connected to a discriminative approach, it is instructive to consider what happens during the last inference iteration of MAP estimation. Once the final set of latent variables $\mathbf{z}^*$ has been determined from Eq. (6.6), the output image $\mathbf{x}^*$ is inferred by maximizing $p(\mathbf{x}|\mathbf{y}, \mathbf{z}^*)$ from Eq. (6.4). This distribution is nothing but an anisotropic (or inhomogeneous) Gaussian random field, whose mean and covariance depend on $\mathbf{y}$ and $\mathbf{z}^*$ (and also $\mathbf{K}$ and σ).

Therefore $\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y},\mathbf{z}^*}$ and $\boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y},\mathbf{z}^*}$ are the mean and covariance parameters of a multivariate normal distribution defined on the whole image, chosen through $\mathbf{z}^*$ so as to hopefully lead to good restoration results. The value of $\mathbf{z}^*$ depends on the specific choice of potential functions via ρ_j and their half-quadratic representations ϕ_j (Eq. 6.6).

It is now natural to ask whether we can instead directly regress the Gaussian random field parameters from the input image. More specifically, we can regress a precision matrix $\boldsymbol{\Theta}(\mathbf{y})$ and a vector $\boldsymbol{\theta}(\mathbf{y})$, leading to $\boldsymbol{\mu} \stackrel{\text{def}}{=} [\boldsymbol{\Theta}(\mathbf{y})]^{-1}\boldsymbol{\theta}(\mathbf{y})$ and $\boldsymbol{\Sigma} \stackrel{\text{def}}{=} [\boldsymbol{\Theta}(\mathbf{y})]^{-1}$. Then the mean $\boldsymbol{\mu}$

and the covariance $\mathbf{\Sigma}$ are learned functions of the observed image $\mathbf{y}$. There are three potential advantages: *First*, we avoid the expensive iterative computation of the half-quadratic optimization. *Second*, we can regress the parameters discriminatively in order to optimize a given performance measure, such as PSNR. *Third*, we are no longer constrained to the form of Eq. (6.6) so that we can now use an expressive regression model on the input image. That is, we are not restricting[2] the resulting model compared to Eq. (6.4); in fact, we can potentially learn a more expressive model.

We arrived at this model from a novel analysis of the half-quadratic approximation, but predicting the means and covariances of a Gaussian model has been done before: Gaussian conditional random fields, first proposed by Tappen et al. [2007], have led to competitive results in image denoising. We build on the more recent regression tree fields (RTFs) by Jancsary et al. [2012a,b].

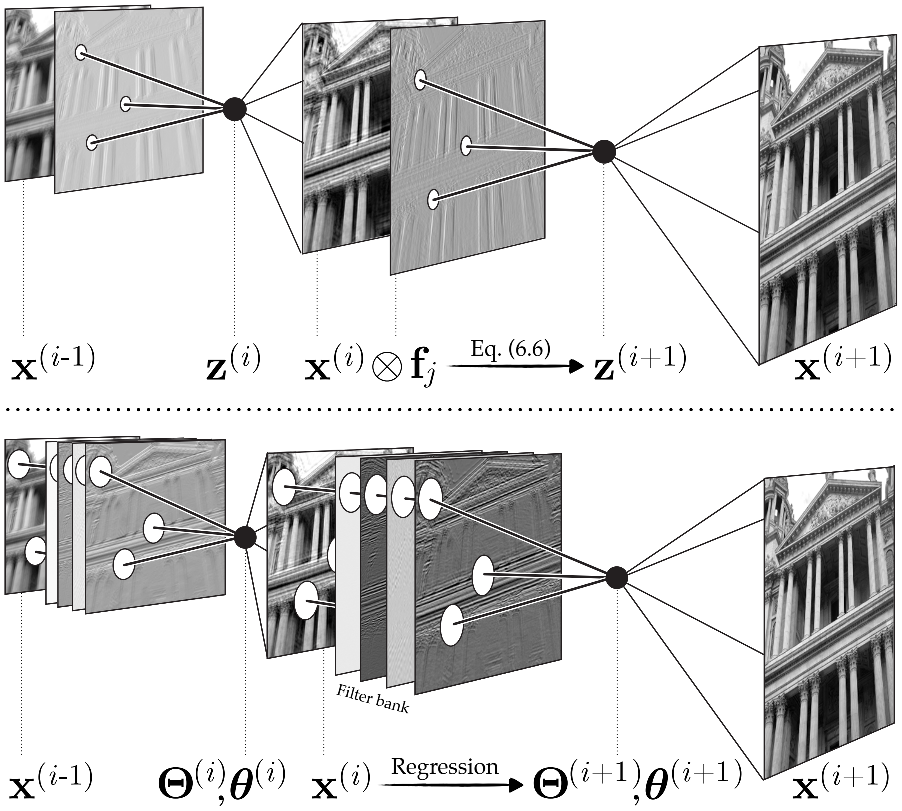
GOING BEYOND DENOISING While such Gaussian CRFs have been successful for image denoising, we argue that applying them to other image restoration applications, such as non-blind image deblurring, is more challenging, since it is difficult to directly regress suitable model parameters. To illustrate this difficulty, let us assume that $\mathbf{f}_j$ are first-order derivative filters. Then, in the generative approach one can think of z_{jc} as modulating pairwise potentials: reducing smoothness constraints in case of large image derivatives *of the output image* $\mathbf{x}$, and imposing smoothness otherwise. In other words, in the generative approach $\mathbf{z}$ determines the local model of the restored image $\mathbf{x}$. Both $\mathbf{x}$ and $\mathbf{z}$ are iteratively refined via half-quadratic inference. In a discriminative model we have access only to the corrupted image $\mathbf{y}$ in order to determine suitable local models.

But in the case of deblurring, the image content in $\mathbf{y}$ is shifted and combined with other parts of the image, depending on an instance-specific blur kernel. This makes the choice of local models difficult. We believe this is one of the reasons why discriminative non-blind deblurring approaches had not been attempted before.

The situation is much easier for image denoising, since it is typically assumed that noise is additive and pixel-independent; hence, one can regress model parameters quite well by averaging values in a neighborhood around a pixel, or more generally by applying a set of filters whose responses provide discriminative features to regress model parameters [*cf.* Tappen et al., 2007; Jancsary et al., 2012a].

---

[2] Note that any multivariate Gaussian distribution can always be expressed as a product of unary and pairwise terms [*cf.* Wainwright and Jordan, 2008], because its exponent is the sum of a linear and quadratic form (*i.e.*, homogeneous polynomials of degrees 1 and 2, respectively). Hence, the final MAP estimate in half-quadratic regularization comes from a pairwise MRF *even if the corresponding sparse image prior models high-order interactions*. This does not mean, however, that high-order dependencies are ignored. They are only hidden in the estimate $\mathbf{z}^*$.



$\mathbf{x}^{(i\text{-}1)}$
$\mathbf{z}^{(i)}$
$\mathbf{x}^{(i)} \otimes \mathbf{f}_j \xrightarrow{\text{Eq. (6.6)}} \mathbf{z}^{(i+1)}$
$\mathbf{x}^{(i+1)}$
Filter bank
$\mathbf{x}^{(i\text{-}1)}$
$\boldsymbol{\Theta}^{(i)}, \boldsymbol{\theta}^{(i)}$
$\mathbf{x}^{(i)} \xrightarrow{\text{Regression}} \boldsymbol{\Theta}^{(i+1)}, \boldsymbol{\theta}^{(i+1)}$
$\mathbf{x}^{(i+1)}$

Figure 6.1: **Standard half-quadratic vs. discriminative cascade.** In a standard half-quadratic approach *(top)*, each z_{jc} can only be updated via Eq. (6.6) based on the filter response $\mathbf{f}_j^\top \mathbf{x}_{(c)}$ of the pixels in the local clique (small white circles, only one filtered image $\mathbf{x} \otimes \mathbf{f}_j$ shown). In the proposed discriminative cascade *(bottom)*, one can use arbitrary features of the image over larger areas (large white circles) to find model parameters $\boldsymbol{\Theta}^{(i)}$ and $\boldsymbol{\theta}^{(i)}$ via regression. At each stage, the functions $\boldsymbol{\Theta}^{(i)}$ and $\boldsymbol{\theta}^{(i)}$ depend on $\mathbf{y}$ through features, such as filter bank responses, image intensities, as well as $\mathbf{x}^{(i)}$ from previous iterations (not shown).

6.1.2 *Discriminative model cascade*

To build a discriminative model for deblurring, we draw inspiration from the iterative refinement of $\mathbf{z}$ in half-quadratic inference. We start with an educated guess of the Gaussian model parameters, regressed from the input image, to obtain a restored image $\mathbf{x}^{(1)}$, which is less corrupted than the original input image. We can then use this as an intermediate result to help regress refined Gaussian model parameters, in order to obtain a better restored image $\mathbf{x}^{(2)}$, *etc.*, effectively obtaining a cascade of refined models. Note that this is a general approach that is not only applicable to image deblurring; other restoration tasks may also benefit from such a model cascade and repeated refinement of the auxiliary variables. As mentioned above, for the special case of image denoising, we can already obtain good parameters at the first model stage and thus obtain a high-quality initial result $\mathbf{x}^{(1)}$ whose restoration quality cannot be improved further as much as for more difficult problems, such as deblurring (*cf.* Section 6.3).

A key advantage of a discriminative approach for predicting model parameters $\boldsymbol{\Theta}^{(i)}$, $\boldsymbol{\theta}^{(i)}$ at the i^{th} stage is its flexibility. As discussed above, a standard generative half-quadratic approach updates each z_{jc} only based on the local clique of the current estimate of the restored image (*cf.* Eq. 6.6). In a discriminative approach, we can regress the parameters based on arbitrary local and global properties of the input image as well as the current estimate of the restored image (see Fig. 6.1 for an illustration). Furthermore, we can use a specialized model (*i. e.*, regression function) for each stage, whereas an image prior in a generative approach does not change during inference. Consequently, we can expect to obtain better estimates in fewer iterations.

OTHER RELATED WORK This iterative procedure can also be linked with earlier ideas about iterative refinement. The idea of *auto-context* [Tu, 2008] is to use the same probabilistic model multiple times in sequence, where each model receives as input the observed image and the output of the previous model in the sequence. Munoz *et al*. train a sequence of (relatively simple) predictors for structured prediction problems, which they call *inference machines* [*e. g.*, Ross et al., 2011]. Loss-specific training of a cascade of discriminative predictors is also related to truncated bi-level optimization (Section 2.4.1, *cf.* [Domke, 2012]). Furthermore, our proposed discriminative cascade is related to the *active random field* of Barbu [2009], which is a multi-stage approach for image denoising that is trained discriminatively. The key difference is that each stage in Barbu [2009] corresponds to a gradient descent iteration of the model energy; moreover, the parameters are shared between all stages.

6.2 GAUSSIAN CRF FOR RESTORATION

As we have seen, a discriminative alternative to half-quadratic MAP estimation is conceptually attractive, but can be challenging due to the need of regressing local image models from the corrupted input image $\mathbf{y}$. To address this challenge we propose a novel Gaussian CRF $p(\mathbf{x}|\mathbf{y};\mathbf{K})$ for image restoration with more challenging Gaussian image corruption models. Let us first consider non-blind image deblurring as a specific example. One challenge in devising such a model is that we cannot train a different model for every blur matrix $\mathbf{K}$; this difficulty may in fact be the reason why previous approaches require separate training for each specific blur kernel [Schuler et al., 2013]. To see how this can be circumvented, we can take inspiration from generative approaches to deblurring and see how the Gaussian blur likelihood $p(\mathbf{y}|\mathbf{x};\mathbf{K})$ contributes to the posterior distribution when assuming a Gaussian prior:

$$\begin{aligned} p(\mathbf{x}|\mathbf{y};\mathbf{K}) &\propto p(\mathbf{y}|\mathbf{x};\mathbf{K}) \cdot p(\mathbf{x}) \\ &\propto \mathcal{N}(\mathbf{y};\mathbf{K}\mathbf{x},\mathbf{I}/\alpha) \cdot \mathcal{N}(\mathbf{x};\boldsymbol{\Theta}^{-1}\boldsymbol{\theta},\boldsymbol{\Theta}^{-1}) \\ &\propto \mathcal{N}\big(\mathbf{x};(\alpha\mathbf{K}^\mathsf{T}\mathbf{K})^{-1}\alpha\mathbf{K}^\mathsf{T}\mathbf{y},(\alpha\mathbf{K}^\mathsf{T}\mathbf{K})^{-1}\big) \cdot \mathcal{N}(\mathbf{x};\boldsymbol{\Theta}^{-1}\boldsymbol{\theta},\boldsymbol{\Theta}^{-1}) \\ &\propto \mathcal{N}\big(\mathbf{x};(\boldsymbol{\Theta}+\alpha\mathbf{K}^\mathsf{T}\mathbf{K})^{-1}(\boldsymbol{\theta}+\alpha\mathbf{K}^\mathsf{T}\mathbf{y}),(\boldsymbol{\Theta}+\alpha\mathbf{K}^\mathsf{T}\mathbf{K})^{-1}\big), \end{aligned} \tag{6.7}$$

where $\alpha = 1/\sigma^2$ relates to the noise level, $\boldsymbol{\Theta}$ is the precision of the Gaussian prior, and $\boldsymbol{\theta}$ relates to its mean. We can now define a Gaussian CRF in which the model parameters $\boldsymbol{\Theta}$ and $\boldsymbol{\theta}$ are not fixed, but regressed from the input image, *i. e.* $\boldsymbol{\Theta} \equiv \boldsymbol{\Theta}(\mathbf{y})$ and $\boldsymbol{\theta} \equiv \boldsymbol{\theta}(\mathbf{y})$ are functions of $\mathbf{y}$. Note that the CRF is parameterized by an instance-specific blur $\mathbf{K}$ as in Eq. (6.7); the blur is *not* used as an input feature to the regressor (although it could be).

Even though motivated through image deblurring, the proposed Gaussian CRF in Eq. (6.7) is not limited to this. Depending on the choice of the matrix $\mathbf{K}$, it can be used to model other applications, such as image super-resolution when $\mathbf{K}$ relates to a downsampling operation. A limitation of this construction is the assumption of Gaussian additive noise, which enables the combination of prior and likelihood terms in closed form.

For the problem of image denoising, *i. e.* $\mathbf{K} = \mathbf{I}$ is an identity matrix, it is worth noting that explicitly incorporating a component related to the likelihood as in Eq. (6.7) may not be necessary, since its contribution could be learned by the regression function. This approach has been pursued by previous work [Tappen et al., 2007; Jancsary et al., 2012b,a] and is also adopted here for the denoising experiments in Section 6.3. It also has the advantage of making no assumption about the type of noise corruption, which allows the removal of non-Gaussian noise, as shown by Jancsary et al. [2012b,a]. In case of deblurring, however, our formulation in Eq. (6.7) does need to make a noise assumption, as a likelihood term is required to adapt the model to arbitrary blurs. But since the regression functions in our discriminative approach do not rely on a particular noise characteristic, our model can still cope with noise that violates the Gaussian assumption to some extent (see Section 6.3).

Once we have determined the parameters via regression, we can obtain a deblurred image as the MAP estimate, which can be derived in closed form as the mean of the Gaussian CRF,

$$\arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y};\mathbf{K}) = (\boldsymbol{\Theta}(\mathbf{y}) + \alpha\mathbf{K}^\mathsf{T}\mathbf{K})^{-1}(\boldsymbol{\theta}(\mathbf{y}) + \alpha\mathbf{K}^\mathsf{T}\mathbf{y}), \tag{6.8}$$

and can be computed by solving a sparse linear system.

OTHER RELATED WORK In recent, independent work, Chen et al. [2013] also combined a discriminatively-trained regularization term with an instance-specific data term for image deblurring and super-resolution. In contrast to our work, they do not provide a formal motivation and do not train the model specifically for these applications. Instead, they train their model for image denoising and then augment it with an instance-specific data term at test time. Furthermore, they cannot combine regularization and data terms in closed form, as they do not use Gaussian random fields. From a different point of view, Cho et al. [2010] propose an adaptive prior for image restoration, which can be seen as a discriminative model whose parameters depend on the observed corrupted image. However, they do not attempt application-specific loss-based training, as we employ here.

6.2.1 *Regression tree field*

To make our approach concrete, we need to specify the regression functions $\boldsymbol{\Theta}(\mathbf{y})$ and $\boldsymbol{\theta}(\mathbf{y})$. To that end, we draw on the recently proposed *regression tree field* (RTF) model by Jancsary et al. [2012a,b]. RTFs have shown state-of-the-art results in image restoration applications, such as image denoising, inpainting, and colorization.

In general, RTFs take the form of a Gaussian CRF in which a non-linear regressor is used to specify the local model parameters. Specifically, regression trees are used, where each leaf stores an individual linear regressor that determines a local potential. Since any Gaussian CRF can be decomposed into factors relating no more than two pixels, our posterior density attains the following form:

$$p(\mathbf{x}|\mathbf{y};\mathbf{K}) \propto \mathcal{N}(\mathbf{y};\mathbf{K}\mathbf{x},\mathbf{I}/\alpha) \cdot \prod_{j=1}^{J+1} \prod_{c \in \mathcal{C}_j} \varphi_j(\mathbf{x}_{(c)},\mathbf{y}) \tag{6.9}$$

$$\varphi_j(\mathbf{x}_{(c)},\mathbf{y}) \propto \exp\Big(-\frac{1}{2}\mathbf{x}_{(c)}^\mathsf{T}\boldsymbol{\Theta}_c^j(\mathbf{y})\mathbf{x}_{(c)} + \mathbf{x}_{(c)}^\mathsf{T}\boldsymbol{\theta}_c^j(\mathbf{y})\Big),$$

where $\mathcal{C}_j$ denotes all pairs of neighboring pixels in the j^{th} of J possible spatial configurations. Concretely, we use 8- and 24-neighborhoods depending on the application and stage in our prediction cascade, *i. e.* $J = 4$ and $J = 12$, respectively (due to spatial symmetries). Additionally, at each stage, we employ a single unary potential via $\varphi_{J+1}(\mathbf{x}_{(c)},\mathbf{y})$, where $\mathcal{C}_{J+1}$ is simply the set of all individual pixels. See Fig. 6.9(a) for an illustration of the neighborhood structure.

We extend previous RTF-based approaches to our setting by *(a)* incorporating the more general Gaussian likelihood if needed, *e. g.* for non-blind image deblurring, as outlined in Eqs. (6.7) and (6.8), and *(b)* by assembling multiple RTFs into a model cascade that iteratively refines the prediction. The cascade will be detailed in Section 6.2.3.

Note that the RTF generalizes the Gaussian CRF of Tappen et al. [2007] in two ways: First, the potentials of an RTF are non-linearly dependent on the input image via non-parametric regression trees.

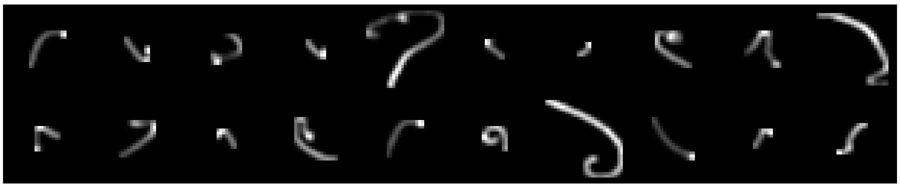


Figure 6.2: **Examples of artificially generated blur kernels.**

Second, the model parameters of arbitrary pairwise Gaussian potentials (with full mean and covariance) are regressed from the input image, whereas Tappen et al. [2007] restrict their parameterization to diagonal weighting of filter responses.

6.2.2 *Training*

While probabilistic training is possible [Jancsary et al., 2012b], we here follow Jancsary et al. [2012a] and learn the regressors using loss-based training, in particular, such that the average *peak signal-to-noise ratio* (PSNR) over all N training images,

$$\mathrm{psnr}(\hat{\mathbf{x}}; \mathbf{x}_{\mathrm{gt}}) = \frac{1}{N} \sum_{i=1}^{N} 20 \log_{10} \left(\frac{R\sqrt{D_i}}{\|\hat{\mathbf{x}}^{(i)} - \mathbf{x}_{\mathrm{gt}}^{(i)}\|} \right) \tag{6.10}$$

is maximized, where D_i denotes the number of pixels in the ground truth image $\mathbf{x}_{\mathrm{gt}}^{(i)}$ and the predicted image $\hat{\mathbf{x}}^{(i)}$ (obtained via Eq. 6.8), and R is the maximum intensity level of a pixel (*e.g.*, $R = 255$). All parameters of the model, including the split functions in the tree and the linear regressors in the leaves, are chosen to maximize PSNR [*cf.* Jancsary et al., 2012a].

Discriminative training necessitates a sufficient amount of training data to ensure generalization. For image denoising, it is easy to synthesize noisy versions of clean ground truth images by adding pixel-independent Gaussian noise (here using standard deviation $\sigma = 25$). We use crops of 256×256 pixels from the BSDS [Arbelaez et al., 2011] as ground truth images. Most image denoising benchmarks (including the one used in our experiments) also consist of synthesized noisy images, hence the training data matches the setting. For image deblurring, supplying appropriate training data is more challenging. Since capturing image pairs of blurred and clean images is difficult, one possible avenue is to also synthesize training data by blurring clean images with realistic blurs. Unfortunately, existing databases [Levin et al., 2009; Köhler et al., 2012] only supply a relatively limited number of blur kernels, and moreover serve also for testing. Hence the model should not be trained on these. We address this problem by generating realistic-looking blur kernels via sampling random 3D trajectories using a simple linear motion model; the obtained trajectories are projected and rasterized to random square kernel sizes in the

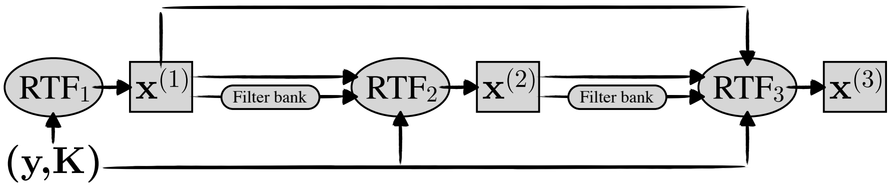


Figure 6.3: **RTF prediction cascade (deblurring).** Only three stages are shown. Cascade similar for denoising, see text for details.

range from 5×5 up to 27×27 pixels (see Fig. 6.2). While it would of course be possible to create even more realistic kernels through more accurate models of camera shake motion[3], we find that these synthetic kernels already allow to generalize well to unseen real blur (*cf.* Section 6.3). We synthetically generate blurred images by convolving each clean image with an artificially generated blur kernel, and subsequently add pixel-independent Gaussian noise (using standard deviations $\sigma = 2.55$ or 0.5, see experiments in Section 6.3). We use crops of 128×128 pixels from the training portion of the BSDS as ground truth images.

6.2.3 *RTF prediction cascade*

IMAGE DEBLURRING As argued in Section 6.1, it is difficult to directly regress good local image models from the blurred input image. Therefore, we employ a cascade of RTF models, where each subsequent model stage uses the output of all previous models as features for the regression (see Fig. 6.3 for an illustration).

We train the first stage of the cascade with minimal conditioning on the input image to avoid overfitting. Concretely, this means the parameters of the unary and pairwise potentials are only linearly regressed from the pixels in the respective cliques (plus a constant pseudo-input, *cf.* [Jancsary et al., 2012a]); we do not use a regression tree. We further use an 8-connected graph structure, resulting in one unary and four pairwise types of potentials (horizontal, vertical, and two diagonals, *cf.* Fig. 6.9(a)). We train this model with 200 pairs of blurred and clean images, which is ample since there are only few model parameters. This model will be referred to as RTF$_1$.

While we do not expect competitive results from RTF$_1$, it is able to remove the dominant blur from the input image (*cf.* Section 6.3 and Fig. 6.8) and makes it much easier for subsequent RTF stages to regress good CRF potentials. Besides the blurred input image, the second stage, RTF$_2$, thus uses the output of RTF$_1$ as an input feature. We additionally evaluate a filter bank on the output of RTF$_1$ to obtain more expressive features. We therein follow [Jancsary et al., 2012a],

---

[3] We think that on average our synthetic blur kernels may in fact be more challenging than typical real ones.

which obtained improved denoising results using the output of a filter bank as input to the regressor. However, we use a different filter bank here, the 16 generatively trained 5×5 filters from the recent FoE model of Gao and Roth [2012]; we found these to outperform other filter banks we have tried, including those used in [Jancsary et al., 2012a].

We use all these features for the split tests in the regression tree (non-linear regression), as well as for the linear potential parameter regressor that is stored in each leaf of the tree. We choose regression trees of depth 7. All subsequent model stages, *i.e.* RTF_3, RTF_4, *etc.*, take as features the outputs from all previous stages, where the filter bank is always only evaluated on the directly preceding model output; see Fig. 6.3 for a schematic. Starting with RTF_2, the Gaussian CRF at each layer uses a 24-connected graph, *i.e.* each pixel is connected to all others in a 5×5 neighborhood. Due to the increased number of model parameters, we train RTF_2 and each subsequent stage with 500 training images, randomly cropped from the training portion of the BSDS and blurred with randomly chosen artificial blur kernels (different at each stage).

IMAGE DENOISING Although it is much easier to directly regress good local image models from a noisy input image, image denoising can also benefit from using a model cascade, as demonstrated in our experimental evaluation (Section 6.3). However, in contrast to our deblurring cascade, we use the same RTF model architecture at each stage, in particular a 24-connected graph structure (5×5 neighborhood), filter bank responses on the output of the directly preceding model stage (or the input image for RTF_1), and regression trees of depth 10. We train each stage with the same 400 training images, cropped from the BSDS. A minor technical difference to our deblurring cascade is that in addition to the original noisy input image, each model stage only uses the output of the directly preceding model stage (*cf.* Fig. 6.3) as feature for the regression (including filter bank responses thereon).

DISCUSSION An interesting property of our model cascade in general is that it yields a restored image after every stage, not only at the end. Even if a deep cascade was trained, at test time we can trade off computational resources versus quality of the restored image by stopping after a certain stage (*cf.* Fig. 6.8; see [Fröhlich et al., 2012] for a segmentation approach that can also be stopped at intermediate stages).

The cascade architecture has another advantage: because each stage in the cascade has access to both the original input image as well as the output of the previous cascade stage, each stage of the cascade enlarges the learning capacity of the overall system. Our cascade ar-

Method	PSNR	Stage	PSNR
KSVD [Elad and Aharon, 2006]	28.28	RTF_1	28.24
5×5 FoE [Gao and Roth, 2012]	28.40	RTF_2	28.62
BM3D [Dabov et al., 2007b]	28.56	RTF_3	28.70
LSSC [Mairal et al., 2009]	28.70	RTF_4	28.74
EPLL [Zoran and Weiss, 2011]	28.68	RTF_5	28.75
opt-MRF [Chen et al., 2013]	28.66		
MLP [Burger et al., 2012]	28.85		

Table 6.1: Average PSNR (dB) on 68 images from [Roth and Black, 2009] for image denoising with $\sigma = 25$ (not quantized); except result of [Burger et al., 2012], left part reproduced from [Chen et al., 2013]. On the right, each row shows the results from the respective stage of our model.

Method	PSNR	Stage	PSNR
3×3 FoE [Schmidt et al., 2010]	27.90	RTF_1	28.25
BLS-GSM [Portilla et al., 2003]	27.98	RTF_2	28.61
5×5 FoE [Gao and Roth, 2012]	28.22	RTF_3	28.69
LSSC [Mairal et al., 2009]	28.23	RTF_4	28.73
BM3D [Dabov et al., 2007b]	28.31	RTF_5	28.74

Table 6.2: Average PSNR (dB) on 68 images from [Roth and Black, 2009] for image denoising with $\sigma = 25$ (8-bit quantized). On the right, each row shows the results from the respective stage of our model.

chitecture therefore provides *nested model classes*, as used in structural risk minimization [Vapnik and Chervonenkis, 1974].

6.3 EXPERIMENTS[4]

6.3.1 *Image denoising*

We first evaluate our approach for image denoising, with a model architecture at each stage that is comparable to that of Jancsary et al. [2012a]. In contrast to [Jancsary et al., 2012a], however, we *(a)* use a model cascade, and *(b)* choose the established denoising benchmark of 68 grayscale images from Roth and Black [2009] (which do not contain images used for training our models). The main aim of these experiments is to demonstrate that a model cascade is beneficial, even for the (comparatively) simpler task of image denoising.

While the denoising results of Jancsary et al. [2012a] could not reach state-of-the-art performance without incorporating the results

---

[4] Code for inference and learning is available on the author's webpage.

Model	PSNR				
CBM3D [Dabov et al., 2007a]	30.18				
Ours:	RTF_1	RTF_2	RTF_3	RTF_4	RTF_5
Channel-independent	28.20	28.55	28.64	28.67	28.68
Channels jointly	30.01	**30.57**			

Table 6.3: Average PSNR (dB) on 68 images (color versions of those used by [Roth and Black, 2009]) for color image denoising with $\sigma = 25$ (added channel-independently, 8-bit quantized).

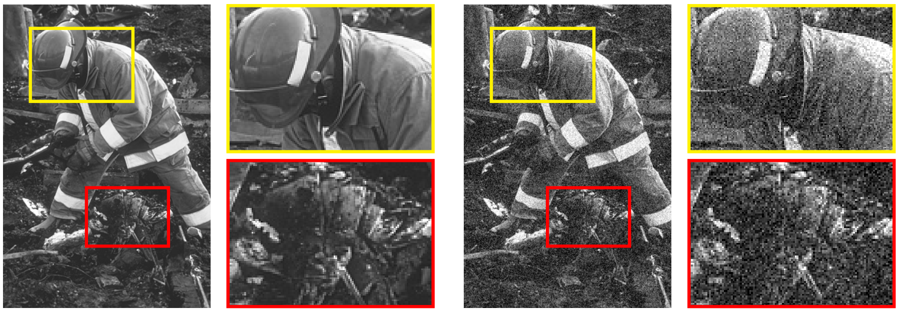
of other denoising methods such as BM3D [Dabov et al., 2007b] as features for the regression trees, our RTF prediction cascade achieves state-of-the-art performance using only the input image (and derived features via the given filter bank). Tab. 6.1 shows that the third model stage RTF_3 is already on par with the second-best competitor LSSC [Mairal et al., 2009], while additional stages further improve performance marginally; the biggest performance improvement is achieved at the second stage. Our model is only outperformed by the neural network of Burger et al. [2012], who trained a multilayer perceptron (MLP) with millions of parameters to denoise image patches. In contrast to our model cascade, their MLP was trained on a huge database of 362 million training examples, which required about a month of training time on a GPU.

While Jancsary et al. [2012a] trained and tested their model without quantizing the images after adding synthetic noise, we additionally considered 8-bit quantized noisy images, *i.e.* image intensity values are rounded and range-limited, *i.e.* in $[0, \ldots, 255]$, as they would be in commonly-used image formats. Repeating the same experiment for 8-bit quantized images shows that we achieve virtually identical results (Tab. 6.2), while the performance of all competing methods deteriorates (often substantially, up to 0.47dB for LSSC). This highlights a strength of the RTF model, which does not make any noise assumptions[5] and can therefore easily deal with the additional quantization noise. A denoising example is shown in Fig. 6.4, which also compares the results of the first and last stage of our prediction cascade.

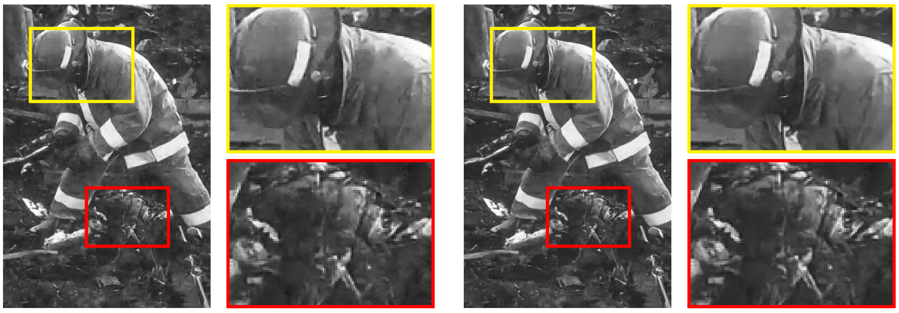
COLOR IMAGE DENOISING As an additional test, we trained a two-stage RTF cascade for color image denoising. To that end, we use the same basic model architecture as for grayscale denoising, but with color triples as input and output of each stage and using the original RGB color images from the Berkeley segmentation dataset. We do not make an attempt to use a realistic color noise model, but instead add Gaussian noise to each color channel independently (fol-

---

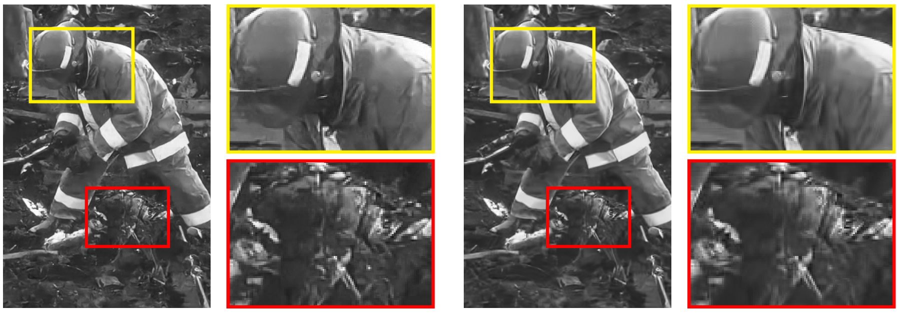
[5] This only applies to our image denoising experiments, where we do not incorporate a likelihood component as we do for image deblurring (*cf.* Section 6.2).



(a) Ground truth (b) Noisy, 20.36dB



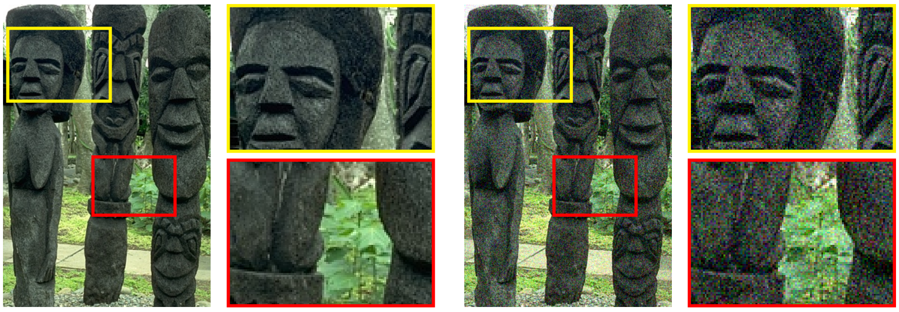
(c) RTF$_1$, 27.04dB (d) RTF$_2$, 27.26dB



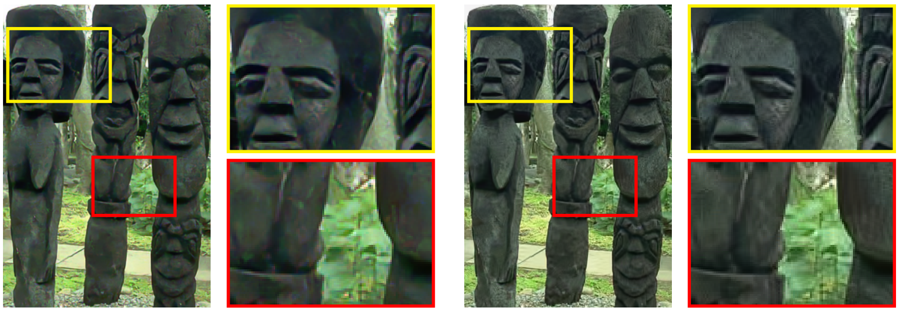
(e) RTF$_5$, 27.33dB (f) BM3D [Dabov et al., 2007b], 26.80dB

Figure 6.4: **Image denoising example (cropped).** While the result of the first stage RTF$_1$ *(c)* is already quite good, it can further be improved by additional stages of our model cascade *(d,e)*, both in terms of PSNR and also visually, where noise in smooth regions is further reduced (such as the firefighter's clothes), while at the same time not oversmoothing textured regions, *e. g.* the rubble at the bottom of the image (which happens for BM3D *(f)*). *Best viewed on screen.*

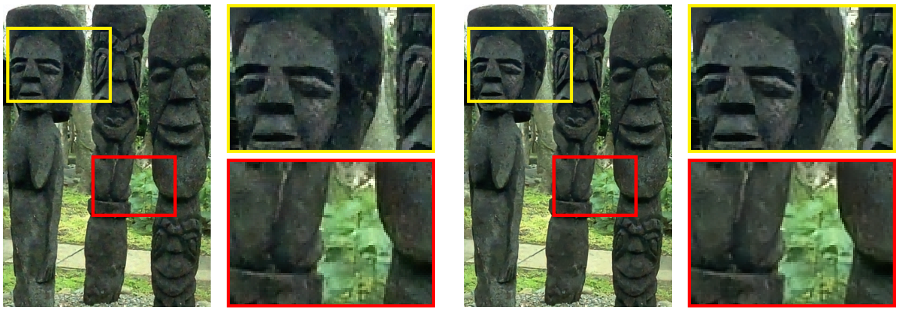
lowed by 8-bit quantization). This experiment aims to show that the RTF can easily exploit correlations between the color channels, and that a model cascade is also beneficial in this case. We employ the same 68 benchmark images, but use the original color images and versions with synthetic noise, generated as described above. Comparing the performance of our dedicated color-denoising RTF cascade to using our grayscale-denoising RTF cascade independently for each channel (for R, G, and B color channels) reveals its superior-



(a) Ground truth (b) Noisy, 20.55dB



(c) Grayscale RTF$_5$, 25.60dB (d) CBM3D [Dabov et al., 2007a], 27.48dB



(e) Color RTF$_1$, 27.57dB (f) Color RTF$_2$, 27.79dB

Figure 6.5: **Color denoising example (cropped).** The trained RTF cascade for color denoising *(e,f)* leads to better quantitative (PSNR) and qualitative results, as compared to applying a model cascade (trained for grayscale image denoising) independently for R, G, and B color channels *(c)*. Correlations between the color channels are exploited to avoid oversmoothing and color artifacts (*cf.* *(c)*). Our results *(e,f)* are competitive with the color denoising method CBM3D *(d)*. *Best viewed on screen.*

ity (Tab. 6.3). It outperforms the baseline grayscale model strongly by about 1.9dB PSNR, even after only two model stages. Furthermore, we outperform the dedicated color denoising approach CBM3D [Dabov et al., 2007a]. Without 8-bit quantization, CBM3D achieves a PSNR of 30.68dB, whereas we might expect a similar performance level of our model as in the case of quantized values (*cf.* Tabs. 6.1 and 6.2). Fig. 6.5

	σ			σ	
Method	2.55	7.65	Stage	2.55	7.65
Lucy-Richardson [Lucy, 1974; Richardson, 1972]	25.38	21.85	RTF$_1$	26.33	24.23
pairw. MRF (MAP) [Krishnan and Fergus, 2009]	26.97	24.91	RTF$_2$	28.21	25.54
2×2 MRF (MAP) [Levin et al., 2007]	28.03	25.36	RTF$_3$	28.50	25.75
5×5 FoE (MAP) [Roth and Black, 2009]	28.44	25.66	RTF$_4$	28.58	25.81
pairw. MRF (MMSE) [Schmidt et al., 2011]	28.24	25.63	RTF$_5$	28.65	25.87
3×3 FoE (MMSE) [Schmidt et al., 2011]	28.66	25.68	RTF$_6$	**28.67**	**25.89**

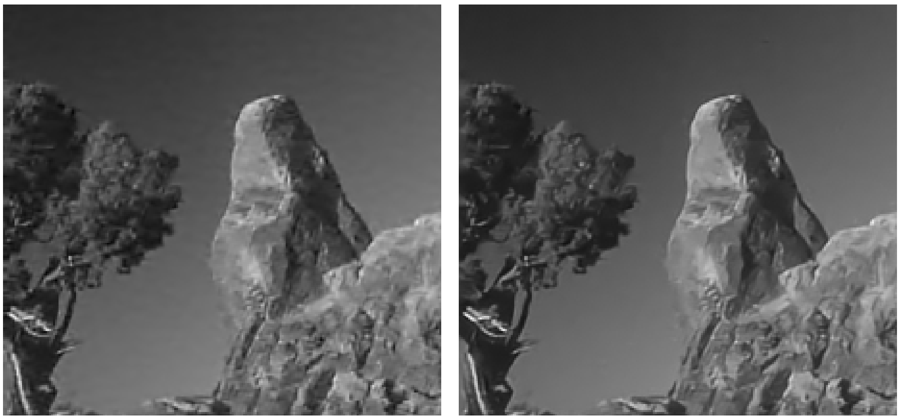
Table 6.4: Average PSNR (dB) on 64 images from [Schmidt et al., 2011] for image deblurring with two noise levels. Left half reproduced from [Schmidt et al., 2011] for ease of comparison.

shows results of our two model cascades applied to color denoising and also compares with CBM3D.

6.3.2 *Image deblurring*

To demonstrate the performance of our approach for the more difficult problem of image deblurring, we apply it to three challenging datasets, specifically to highlight individual benefits. First, we analyze the performance in the typical evaluation scenario for non-blind deblurring, *i. e.* when training and testing is carried out with (nearly) perfect blur kernels. Second, we evaluate the generalization ability of our approach by training the model to deal with imperfect blur kernels. This is important for blind deblurring, where the estimated blur kernels generally contain some errors. Third, we demonstrate the applicability to realistic images of somewhat higher resolution. Please note that images and kernels are always kept strictly separate for training and testing in all experiments.

STANDARD EVALUATION We trained a six-stage RTF prediction cascade as described in Section 6.2 and evaluate all stages individually on 64 test images taken from [Schmidt et al., 2011], which have also been used in Chapter 4. Training images have been blurred synthetically with 1% additive white Gaussian noise ($\sigma = 2.55$); test images with both $\sigma = 2.55$ and a higher noise level of $\sigma = 7.65$. While we used artificial blur kernels to generate our training data, the test images from [Schmidt et al., 2011] have been created with the realistic kernels from [Levin et al., 2009]. The blur kernels used for deblurring in the benchmark are slightly perturbed from the ground truth to mimic kernel estimation errors (following *e. g.* [Krishnan and Fergus, 2009]), but the perturbation is somewhat minor and does not necessarily reflect typical kernel estimation errors; hence we test a more



(a) [Schmidt et al., 2011], PSNR = 29.05dB (b) RTF_6, PSNR = 29.23dB

Figure 6.6: **Deblurring example (cropped).** Qualitative comparison with the high-quality approach of Schmidt et al. [2011] (3×3 FoE, MMSE estimation, *cf.* Chapter 4). Our approach *(b)* reconstructs smooth and textured areas well, exhibits fewer artifacts, and is many times faster. *Best viewed zoomed in on screen.*

realistic scenario later on (see below). We compare our average PSNR performance to all methods that were evaluated in [Schmidt et al., 2011]. The results in Tab. 6.4 show that we achieve state-of-the-art performance that is on par with the high-quality sampling-based approach of Schmidt et al. [2011] at $\sigma = 2.55$, and even outperforms it at $\sigma = 7.65$ despite not being trained for this noise level (only α was adapted, see Eq. 6.7). As we shall discuss below, our approach is many times faster, however. At the noise level our model is trained for ($\sigma = 2.55$), we strongly outperform the efficient half-quadratic regularization approach of Krishnan and Fergus [2009] by over 1.5dB, and the popular method of Levin et al. [2007] by 0.6dB. The clear performance gains at the higher noise level demonstrate our model's noise generalization. We further notice that the weakly conditional first stage (RTF_1) leads only to modest performance levels here; RTF_2 and RTF_3 boost the performance substantially further. Later stages lead to additional gains, but less so. Aside from the raw numbers, it is noteworthy that our model is able to preserve small details, while at the same time reconstructing smooth areas well (see Fig. 6.6 for an example). Note that this is not the case for the approaches tested in [Schmidt et al., 2011].

This demonstrates that when testing (and training) is done with the correct (*i.e.* ground truth) blur kernels, our approach achieves very good results. Even though we train our model on artificially generated blur kernels (Fig. 6.2), it apparently generalizes well to real blurs.

ADAPTATION TO KERNEL ESTIMATION ERRORS Blind deblurring approaches often produce kernel estimates with substantial errors,

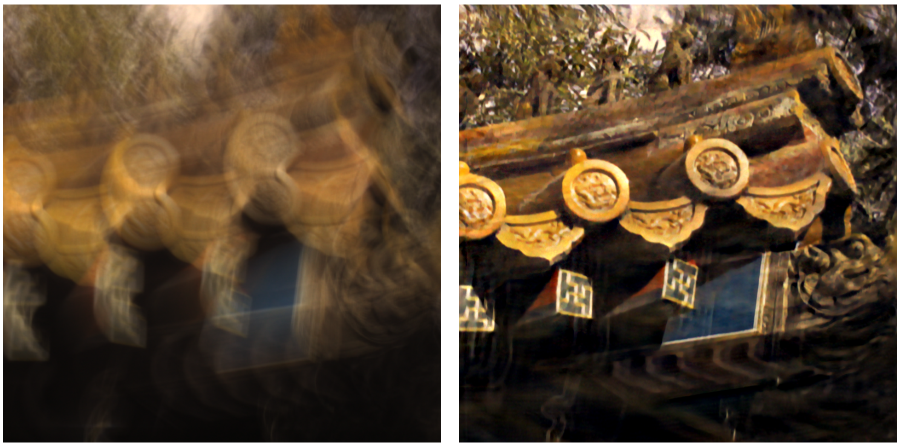


Figure 6.7: Deblurring example from the benchmark of [Köhler et al., 2012] (*cf.* Tab. 6.6), showing the result of our RTF_2 model *(right)* given the blurred image *(left)* and kernel estimate by [Xu and Jia, 2010].

which can cause ringing artifacts in the restored image [*cf.* Yuan et al., 2008]. Hence, it is important to evaluate and adapt our model to this realistic scenario. To train our model for this task, we experimented with adding noise to the ground truth kernels and also used estimated kernels for training.

We consider the data of [Levin et al., 2011] as a benchmark, which provides several kernel estimates besides blurred and ground truth images for 32 test instances, as well as deblurring results with the various kernel estimates. Since the amount of noise in these blurred images is significantly lower than in the benchmark of [Schmidt et al., 2011], we only added Gaussian noise with $\sigma = 0.5$ to our training images. We evaluate average PSNR performance over all 32 images (using code by [Levin et al., 2011] to account for translations in kernel estimates) instead of error ratios as in [Levin et al., 2011], since we are not interested in the quality of the estimated kernels itself, but rather the final restoration performance given the estimated kernels.

The results in Tab. 6.5 show that training with ground truth kernels leads to subpar performance when kernel estimates are used at test time. Adding noise to the ground truth kernels for training leads to improved results of RTF_1 with estimated kernels at test time, but performance of our second stage model RTF_2 already deteriorates; hence those noisy kernels are not an ideal proxy for real kernel estimates. However, we achieve superior results by training our model with a mix of perfect and estimated kernels (obtained with the method of Xu and Jia [2010]), *i.e.* for half of the synthetically blurred training images we use an estimated kernel instead of the ground truth kernel[6]. Compared to the deblurred images from [Levin et al., 2011] (which used the non-blind approach of Levin et al. [2007]), we achieve

---

[6] Here, we trained RTF_1 and RTF_2 with the same 200 images as it was time-consuming to obtain good enough kernel estimates for training.

substantial performance improvements for deblurring with estimated kernels of up to 0.72dB (for kernels from [Fergus et al., 2006]). Furthermore, it is interesting to note that the first stage of our model already achieves good performance; this is presumably due to the much reduced amount of noise in this benchmark[7].

Since the publication of [Schmidt et al., 2013], Schelten et al. [2015] extended our approach to *blind* deconvolution, specifically addressing the issue of kernel estimation errors. Concretely, they start with the blurred image and a blur estimate as we do here, but then alternate between updating the restored image via RTFs and refinement of the blur kernel estimate.

REALISTIC HIGHER-RESOLUTION IMAGES We consider the recent benchmark for camera shake by Köhler et al. [2012] to demonstrate results on realistic images of increased spatial resolution; these images may substantially violate our model's stationary blur and Gaussian noise assumptions (which can deteriorate performance [*cf.* Cho et al., 2011; Tai and Lin, 2012]). The benchmark consists of 4 color images of size 800×800 pixels blurred by 12 different real camera motions, yielding 48 images in total. The overall best performing blind deblurring approach in this benchmark is the one by Xu and Jia [2010] despite making a stationary blur assumption, *i. e.* the same blur kernel is used in all parts of the image. We use the provided kernel estimates by [Xu and Jia, 2010] from the benchmark dataset, but with our non-blind method to obtain the deblurred image (treating color channels R, G, and B independently). Tab. 6.6 shows that performance (evaluated using the provided code) can substantially be improved by using our RTF_2 model instead of their non-blind step (which is related to [Krishnan and Fergus, 2009]). While Xu and Jia's non-blind step is inherently faster, it does lead to substantially worse results, here on average 0.41dB. Fig. 6.7 shows an example of a deblurred image. Note that the RTF_2 model used here is the same as in Tab. 6.5, *i. e.* trained with a mix of ground truth and estimated kernels (using [Xu and Jia, 2010]), and additive Gaussian noise with $\sigma = 0.5$.

RUNTIME The computational demand of our method is comparable to the half-quadratic approach of Levin et al. [2007], but uses this computational budget more effectively due to its discriminative nature (*cf.* Section 6.1 and Fig. 6.1). Also note that the tree-based regressor is very efficient. As a result, we achieve state-of-the-art performance on par with the best result of [Schmidt et al., 2011], but much faster: about 2 seconds per image in Tab. 6.4 (all six model stages combined) compared to 4 minutes for [Schmidt et al., 2011].

---

[7] Theoretically, in the absence of noise, non-blind deblurring can be solved exactly without any regularization by inverting the blur matrix.

Kernels (estimated) for testing	Method						
		Ours with different kernels for training					
	Levin et al. [2007]	GT		GT + Noise		GT & Est.	
		RTF$_1$	RTF$_2$	RTF$_1$	RTF$_2$	RTF$_1$	RTF$_2$
Ground truth (GT)	32.73	32.76	33.81	32.08	30.51	32.90	**33.97**
Levin et al. [2011]	30.05	29.41	29.52	29.73	29.03	29.90	**30.40**
Cho and Lee [2009]	29.71[8]	28.29	27.76	29.36	28.75	29.33	**29.73**
Fergus et al. [2006]	28.38	27.86	27.84	28.49	27.58	28.63	**29.10**
GT + Noise	—	26.67	26.52	28.69	**30.34**	28.10	28.07
Xu and Jia [2010]	—	29.04	28.29	30.25	29.56	30.30	**30.84**

Table 6.5: Deblurring results (PSNR in dB, average over 32 images from [Levin et al., 2011]) that analyze the ability to cope with kernel estimation errors. The kernel estimates of [Levin et al., 2011; Cho and Lee, 2009; Fergus et al., 2006] are provided by [Levin et al., 2011]; the kernel estimates of [Xu and Jia, 2010] are obtained using the authors' code. In the last two columns, "GT & Est." refers to using a mix of ground truth and estimated kernels (via [Xu and Jia, 2010]) for training. The second column shows the average performance of the non-blind method of Levin et al. [2007] for various kernels, as provided by [Levin et al., 2011]. For the kernel estimates of [Levin et al., 2011] (6th row), we used the "free energy with diagonal covariance approximation" algorithm in the filter domain.

For the benchmark in Tab. 6.5 with larger images, we require around 3 seconds for each model stage.

6.4 DISCUSSION

6.4.1 *Training dataset*

For image denoising (Tabs. 6.1 and 6.2) and image deblurring in typical evaluation scenarios (*i. e.* true blur kernel and noise level known at test time, Tab. 6.4), we have trained RTF model cascades for up to six stages, with each additional stage improving restoration performance (although with diminishing improvements in the later stages). However, this does not apply to deblurring in the context of blind deblurring, *i. e.* where erroneous estimated blur kernels are used at test time. Especially under realistic conditions (Tab. 6.6), the blur might be spatially varying and the noise may not be Gaussian. Under these conditions, it is much more difficult to find a suitable training set in

---

[8] This result taken from [Levin et al., 2011] may have employed the non-blind method from [Cho and Lee, 2009].

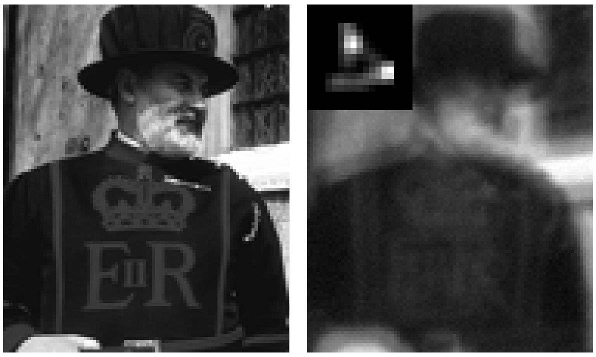
Kernel	Image 1	Image 2	Image 3	Image 4
1	+0.44	+0.54	+1.05	+0.76
2	+0.44	+0.27	+0.38	+0.46
3	+0.02	+0.03	+0.39	−0.26
4	+0.31	+0.30	+0.61	+0.27
5	+0.61	+0.44	+0.64	+0.05
6	+0.40	+0.41	+1.03	+0.48
7	+0.24	+0.55	+0.45	+0.31
8	+0.76	+0.56	+2.17	+1.73
9	+0.35	−0.09	+0.02	+0.23
10	+0.19	−0.55	+0.25	+0.29
11	−0.19	−0.43	+0.46	+0.09
12	+0.76	+0.04	+0.66	+0.64

Table 6.6: Performance gain (PSNR in dB) over the results of [Xu and Jia, 2010] in the benchmark of [Köhler et al., 2012] for each combination of 4 test images and 12 blur kernels. We use the provided blur kernel estimates of [Xu and Jia, 2010] with our RTF_2 model for non-blind deblurring. We can improve the performance in 43 of 48 test instances, on average about 0.41dB.

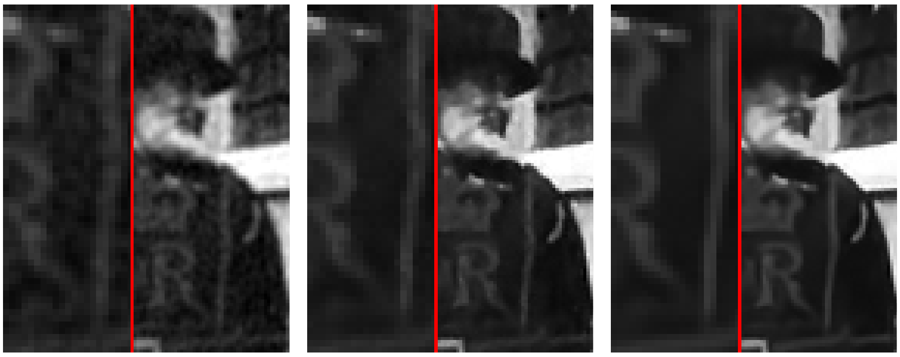
a discriminative setting such as ours. We initially tried using noisy blur kernels as a proxy for estimated kernels at test time, but only achieved performance improvements at the first model stage (*cf*. Tab. 6.5); in fact it was challenging to learn a second model stage that would improve upon the first. While we showed it to be possible to outperform existing approaches by training our model with a mix of ground truth and estimated kernels (*cf*. Section 6.3.2), we believe substantially improved results could be achieved with training datasets that more closely match the conditions encountered at test time. Future work should thus aim to provide realistic data with ground truth also for training discriminative approaches [*e.g.*, Schelten et al., 2015], not only for benchmarking.

6.4.2 *Model connectivity and comparison*

Random field models for image restoration typically use (manually defined) pairwise connectivity (4-connected neighborhood, *i.e.* horizontal and vertical direct neighbor), or alternatively follow the Field of Experts (FoE) framework [Roth and Black, 2009], which models responses of a (learned) filter bank of extended size (5×5 often used, see Fig. 6.9(b) for an example). In contrast, the regression tree field, as introduced by Jancsary et al. [2012b] and also used here, employs learned (and possibly long-range) pairwise connections; see Fig. 6.9(a) for an illustration. In a Gaussian random field, such as the RTF, all



(a) Ground truth (b) Blurred, 15.68dB

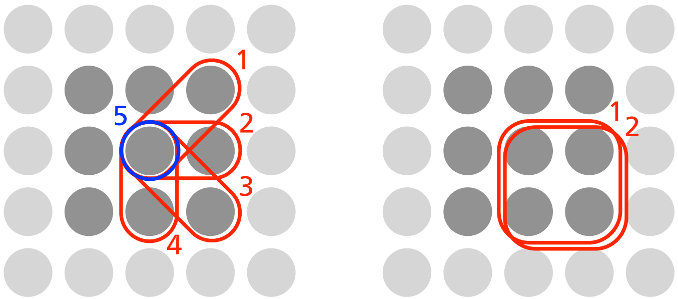


(c) RTF_1, 25.39dB (d) RTF_2, 27.71dB (e) RTF_6, 28.20dB

Figure 6.8: **Deblurring example at different model stages.** The first stage RTF_1 removes dominant blur from the image *(c)*, but artifacts remain. The second stage RTF_2 *(d)* substantially improves upon this result quantitatively (PSNR in dB) and qualitatively. Further model stages continue to suppress noise and refine image details *(e)*. The left sides of *(c–e)* show a closeup view of image details on the respective right sides. The blur kernel is shown at the upper left of *(b)*, scaled and resized for better visualization. *Best viewed on screen.*

high-order factors can always be expressed through pairwise ones [Wainwright and Jordan, 2008]. Hence, no modeling power is lost by restricting factors to pairwise (and unary) connectivity.

In Fig. 6.9, both RTF and FoE are shown with 8-connected neighborhoods, *i. e.* the central pixel is connected to its nearest 8 neighbors (depicted in dark gray). We have used a 24-connected neighborhood in most RTF model stages. An identical connectivity is achieved via a Field of Experts model with 3×3 filters. In general, an FoE model with filters of size $m \times m$ yields a $2m^2 - 1$ neighborhood connectivity. In an RTF, denser connectivity can be achieved by adding more long-range pairwise connections, but this becomes prohibitively expensive to train in the current setting, where training complexity is linear in the number of factor types. Of course, one could modify the RTF to also model filter responses, which may be the subject of future work.



(a) Regression tree field (b) Field of experts

Figure 6.9: **Factor types for 8-connected random fields (shown anchored at central pixel).** *(a)* RTF with four pairwise (red) and one unary (blue) factor type, and *(b)* filter-based random field model (FoE [Roth and Black, 2009]) with two filters of size 2×2 (red).

6.5 SUMMARY

From a novel analysis of common half-quadratic inference, we introduced a discriminative image restoration approach, applicable to image restoration problems that can be expressed through (arbitrary) quadratic data terms. While inspired by half-quadratic regularization, our approach offers a generalization that does not separate between model and inference anymore, as in traditional HQ methods.

We enable discriminative prediction in the context of challenging Gaussian image corruption models by separating the instance-specific parameters of the data model from the discriminative parameter regression, which for deblurring allows coping with arbitrary blur kernels at test time without needing to retrain the model. Moreover, a discriminative prediction cascade helps to overcome the problem of regressing suitable parameters directly from the input image. Our proposed cascade model is based on regression tree fields at each stage, which are trained by loss minimization on training data generated according to the given data term.

We demonstrated its merit for image denoising and especially for the problem of non-blind deblurring. For deblurring, we employed synthesized blur kernels to generate training data. We demonstrated state-of-the art performance on several challenging benchmarks, including robustness to kernel estimation errors in the context of blind deblurring. Our approach is not limited to image denoising and deblurring, and can be extended to other image restoration applications, especially when their data term takes a quadratic form.

7

DEEP SHRINKAGE FIELDS FOR EFFECTIVE IMAGE RESTORATION

CONTENTS

7.1 Related work 156
7.2 Half-quadratic Baseline 157
  7.2.1 Shrinkage function 160
7.3 Cascade of Shrinkage Fields 162
  7.3.1 Learning 163
7.4 Experiments 165
7.5 Summary 170

Image restoration methods for removing imaging artifacts, such as noise, blur, moiré *etc*. have received significant attention in both academic research, as well as in practical applications of digital imaging [*e. g.*, DxO Image Science, 2013]. In academic research, the focus has been predominantly on achieving utmost image quality, largely disregarding the computational effort of the restoration process [Roth and Black, 2009; Zoran and Weiss, 2011; Mairal et al., 2009]. In practical digital imaging, the computational resources are often severely constrained, however, since the processing capacity of on-camera hardware is many times lower than that of a conventional desktop PC. But even on a desktop PC state-of-the-art techniques often take minutes to denoise a small VGA-sized image (equivalent to 0.3 megapixels). Modern digital cameras take images of 16 and more megapixels, on the other hand, to which existing techniques by and large do not scale. The main notable exception is BM3D [Dabov et al., 2007b], which offers high efficiency and image quality, but is a heavily engineered method with years of refinement. Moreover, its use of block matching as the key computational component makes an implementation on parallel architectures, such as GPUs and DSPs, challenging. One may hope that advances in embedded hardware will make the direct on-camera usage of existing advanced restoration techniques possible in the future, but it is not unlikely that the image resolution will increase as well. Consequently, to bridge the existing gap in computational efficiency of image restoration techniques and at the same time achieve high image quality, a different image restoration approach is needed.

In this chapter we introduce *shrinkage fields*, a principled image restoration architecture that is derived from existing optimization algorithms for common random field models. In particular, shrinkage fields owe their computational efficiency to a specific kind of quadratic relaxation technique that is derived from the additive form of half-quadratic (HQ) optimization (Section 3.4.2) – the only operations not applied at a per-pixel level are convolutions and discrete Fourier transforms (DFTs). But unlike existing additive HQ approaches [Krishnan and Fergus, 2009; Wang et al., 2008], we make full use of learning through loss-based training with application-specific loss functions [*cf.* Jancsary et al., 2012a], which allows us to achieve higher levels of restoration quality. Moreover and in contrast to standard random fields, which are specified through potential functions, shrinkage fields model the "shrinkage functions" associated with the potential directly. This increases the flexibility over half-quadratic approaches of the additive form, since we can show that potential functions always lead to monotonic shrinkage functions. In contrast, we can – and do – learn non-monotonic shrinkage functions, similar to those that have been discriminatively learned in the context of wavelet image denoising [Hel-Or and Shaked, 2008]. More importantly, using shrinkage functions directly admits efficient learning, because the model prediction and its gradient w.r.t. the model parameters can be computed in closed form. Finally, our approach employs a prediction cascade (Chapter 6), using multiple model stages for iterative refinement. Loosely speaking, we learn the random field and the iterative optimization algorithm at the same time [*cf.* Barbu, 2009].

The proposed approach has several key benefits: *(1)* It is conceptually simple and derived from standard inference procedures for random field models; *(2)* it achieves very high levels of image quality on par with, or surpassing, the current state of the art; *(3)* it is computationally very efficient with a complexity of $\mathcal{O}(D \log D)$ (where D is the number of pixels); *(4)* it offers high levels of parallelism making it well suited for GPU or DSP implementations; *(5)* unlike heavily engineered techniques, such as BM3D, all parameters can be directly learned from example data using simple gradient-based optimization, making it easy to apply and adapt to new settings, such as different trade-offs between efficiency and restoration quality.

7.1 RELATED WORK

The connection between regularization or priors and shrinkage functions has been widely studied in wavelet image restoration [*e.g.*, Simoncelli, 1999; Antoniadis and Fan, 2001]. We mentioned the connection between the additive form of half-quadratic optimization and shrinkage functions (as proximal operators) in Section 3.4.2.5, which has also been noted by Wang et al. [2008]. Based on the approach of Wang et al. [2008], Krishnan and Fergus [2009] popularized additive half-quadratic optimization for the task of non-blind deconvolution [*e.g.*, Xu and Jia, 2010]. We start from this connection here, but in contrast do not use fixed potential functions, but employ more general, learned shrinkage functions.

Discriminative training of continuous conditional random fields (CRFs) for image restoration has been proposed by Samuel and Tappen [2009], which has recently been revisited by Chen et al. [2013]. Gaussian CRFs and their associated loss-based training have first been introduced by Tappen et al. [2007]. Recently, Jancsary et al. [2012a] improved upon this by introducing regression tree fields (RTFs), a more flexible Gaussian CRF that is also trained by loss minimization. In contrast to these previous approaches, the proposed shrinkage fields admit more efficient inference and can be trained very easily by means of standard gradient-based optimization. Discriminatively learning a random field model and its associated optimization algorithm has been proposed by Barbu [2009]. In the same vein, we trained a cascade of RTFs in Chapter 6 [*cf.* Schmidt et al., 2016]. While [Barbu, 2009] is very efficient, it yields lower image quality given the same model complexity, and relies on a complicated and time-consuming learning procedure. Our approach is conceptually most similar to Chapter 6, which is *generally* motivated as an extension of half-quadratic inference. However, here we additionally derive the model parameterization (shrinkage functions for filter responses) *specifically* as an extension of the additive half-quadratic form. By doing so, we trade-off modeling flexibility (compared to Chapter 6) against far more efficient inference and ease of training.

7.2 HALF-QUADRATIC BASELINE

As a starting point we consider restoring an image $\mathbf{x}$ from its corrupted observation $\mathbf{y}$ by combining an observation likelihood and an image prior invoking Bayes' rule:

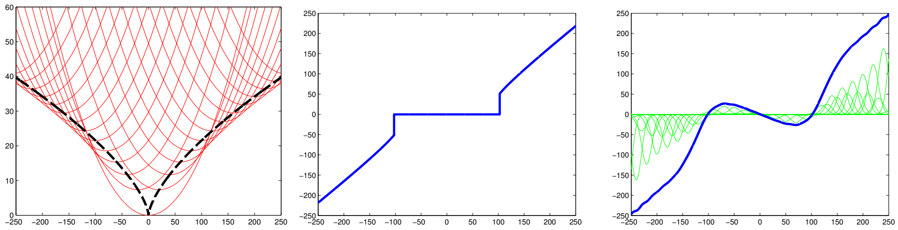
$$p(\mathbf{x}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}) \cdot p(\mathbf{x}) \tag{7.1}$$

$$\propto \mathcal{N}(\mathbf{y}; \mathbf{K}\mathbf{x}, \mathbf{I}/\lambda) \cdot \prod_{i=1}^{N} \prod_{c \in \mathcal{C}} \exp\left(-\rho_i(\mathbf{f}_i^{\mathsf{T}} \mathbf{x}_{(c)})\right). \tag{7.2}$$

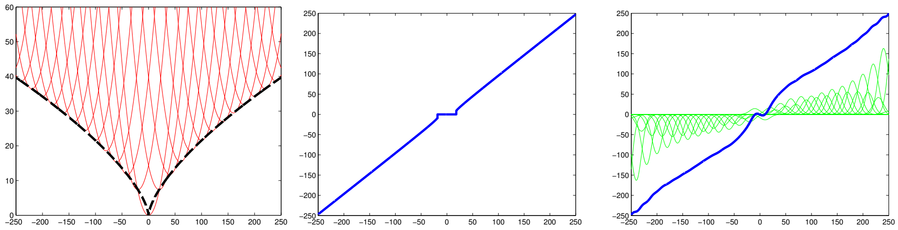
The corruption process is as usual modeled with a Gaussian likelihood (or data term), where $\mathbf{K}\mathbf{x} \equiv \mathbf{k} \otimes \mathbf{x}$ denotes convolution of $\mathbf{x}$ with a kernel (point spread function) $\mathbf{k}$, and λ is related to the strength of the assumed additive Gaussian noise. Regularization is provided through a Field of Experts (FoE) [Roth and Black, 2009] with robust potential functions $e^{-\rho_i}$ that model the responses $\mathbf{f}_i^{\mathsf{T}} \mathbf{x}_{(c)}$ of filters $\mathbf{f}_i$ over all cliques $c \in \mathcal{C}$ of the image $\mathbf{x}$.

The posterior distribution $p(\mathbf{x}|\mathbf{y}) \propto \exp\left(-E(\mathbf{x}|\mathbf{y})\right)$ can be expressed by its associated Gibbs energy

$$E(\mathbf{x}|\mathbf{y}) = \frac{\lambda}{2} \|\mathbf{y} - \mathbf{K}\mathbf{x}\|^2 + \sum_{i=1}^{N} \sum_{c \in \mathcal{C}} \rho_i(\mathbf{f}_i^{\mathsf{T}} \mathbf{x}_{(c)}), \tag{7.3}$$



(a) $\rho(v), \beta = 0.0035$ (b) $f_\beta(v), \beta = 0.0035$ (c) Stage 1 of csf$_{\text{pw.}}$



(d) $\rho(v), \beta = 0.035$ (e) $f_\beta(v), \beta = 0.035$ (f) Stage 2 of csf$_{\text{pw.}}$

Figure 7.1: *(a,d)* Penalty $\rho(v) = |v|^{2/3}$ (dashed, black) and its quadratic relaxation $\rho(z) + \frac{\beta}{2}(v - z)^2$ for some values of z (solid, red). *(b,e)* Associated shrinkage function $f_\beta(v) = \arg\min_z \left(\rho(z) + \frac{\beta}{2}(v - z)^2\right)$ for $\rho(z) = |z|^{2/3}$ and given β. *(c,f)* Learned shrinkage functions $f_\pi(v) = \sum_{j=1}^{M} \pi_j \exp\left(-\frac{\gamma}{2}(v - \mu_j)^2\right)$ (solid, blue) of csf$_{\text{pw.}}$ (*cf.* Section 7.4) as combination of Gaussian RBF kernels (solid, green).

which allows to predict the restored image in case of MAP estimation by finding $\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}) = \arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$.

One way to minimize Eq. (7.3) is to directly employ gradient-descent algorithms. Another popular approach is HQ inference (Chapter 3), which we analyze and extend here. Recall that for half-quadratic MAP estimation (of the envelope type), we first introduce independent auxiliary variables z_{ic} for all filter responses $\mathbf{f}_i^{\mathsf{T}}\mathbf{x}_{(c)}$ to obtain an augmented energy $E(\mathbf{x}, \mathbf{z}|\mathbf{y})$ in such a way that $\arg\min_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}) = \arg\min_{\mathbf{x},\mathbf{z}} E(\mathbf{x}, \mathbf{z}|\mathbf{y})$. A block coordinate descent strategy (Alg. 3.1), which alternates between minimizing w.r.t. $\mathbf{x}$ and $\mathbf{z}$, is then used to minimize $E(\mathbf{x}, \mathbf{z}|\mathbf{y})$. Each iteration of the algorithm uses a different quadratic bound $E(\mathbf{x}|\mathbf{z}, \mathbf{y})$ of the original objective function $E(\mathbf{x}|\mathbf{y})$, determined by auxiliary variables $\mathbf{z}$. This approach typically has faster convergence than minimizing $E(\mathbf{x}|\mathbf{y})$ directly, and each descent step is often relatively simple to carry out. That is because auxiliary variables are introduced in such a way that $E(\mathbf{x}|\mathbf{z}, \mathbf{y})$[1] becomes a quadratic function; minimizing $E(\mathbf{z}|\mathbf{x}, \mathbf{y})$ simply amounts to solving many independent univariate optimization problems.

Recall that we further categorized HQ approaches into *additive* (Section 3.4.2, [Geman and Yang, 1995]) and *multiplicative* (Section 3.4.1, [Geman and Reynolds, 1992]) forms. A main computational differ-

---

[1] $p(\mathbf{x}|\mathbf{z}, \mathbf{y}) \propto \exp\left(-E(\mathbf{x}|\mathbf{z}, \mathbf{y})\right)$, other energies defined accordingly.

Algorithm 7.1 Half-quadratic minimization with continuation

```
Require: $\beta$-schedule $\beta_1, \dots, \beta_T$ with $\beta_{t+1} > \beta_t$
  $\hat{\mathbf{x}}_0 \leftarrow \mathbf{y}$
  for $t \leftarrow 1$ to $T$ do
      $\hat{z}_{ic} \leftarrow \arg\min_{z_{ic}} E_{\beta_t}(\mathbf{z}|\hat{\mathbf{x}}_{t-1}, \mathbf{y}) = f_{i,\beta_t}(\mathbf{f}_i^\mathsf{T} \hat{\mathbf{x}}_{t-1(c)})$
      $\hat{\mathbf{x}}_t \leftarrow \arg\min_{\mathbf{x}} E_{\beta_t}(\mathbf{x}|\hat{\mathbf{z}}, \mathbf{y}) = g_{\beta_t}(\hat{\mathbf{z}})$
```

ence in practice is that $\arg\min_{\mathbf{x}} E(\mathbf{x}|\mathbf{z},\mathbf{y}) = \boldsymbol{\Omega}(\mathbf{z},\mathbf{y})^{-1}\boldsymbol{\eta}(\mathbf{y})$ in the multiplicative form, and $\arg\min_{\mathbf{x}} E(\mathbf{x}|\mathbf{z},\mathbf{y}) = \boldsymbol{\Omega}(\mathbf{y})^{-1}\boldsymbol{\eta}(\mathbf{z},\mathbf{y})$ in the additive form. Here, $\boldsymbol{\Omega} \in \mathbb{R}^{D \times D}$ is a sparse matrix with D being the number of pixels, and $\boldsymbol{\eta} \in \mathbb{R}^D$ is a vector. That implies that the quadratic function can be minimized by solving a system of linear equations, where in the multiplicative form, $\mathbf{z}$ only influences the equation system matrix $\boldsymbol{\Omega}$, and in the additive form only the right-hand side $\boldsymbol{\eta}$ of the equation system. Hence, the additive form is in general computationally more attractive since the equation system matrix stays constant during iterative optimization (*e. g.*, a factorization of $\boldsymbol{\Omega}$ could be re-used, or $\boldsymbol{\Omega}$ might be diagonalized with a change of basis).

However, a challenge is that the additive form is not directly applicable to many heavy-tailed potential functions of practical relevance, since the associated penalty functions ρ are often not smooth enough (*cf.* Section 3.4.2.5). To remedy this and to speed up convergence, Wang et al. [2008] proposed a continuation scheme, where a parameter β is increased during the half-quadratic optimization (*cf.* Alg. 7.1). Concretely, as discussed in Section 3.4.2.5, the problem is cast as $\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} E(\mathbf{x}|\mathbf{y}) = \arg\min_{\substack{\mathbf{x},\mathbf{z} \\ \beta \to \infty}} E_\beta(\mathbf{x},\mathbf{z}|\mathbf{y})$ with

$$E_\beta(\mathbf{x},\mathbf{z}|\mathbf{y}) = \frac{\lambda}{2}\|\mathbf{y} - \mathbf{K}\mathbf{x}\|^2 + \sum_{i=1}^{N}\sum_{c \in C}\left(\frac{\beta}{2}\left(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)} - z_{ic}\right)^2 + \rho_i(z_{ic})\right). \tag{7.4}$$

Intuitively, when $\beta \to \infty$, the auxiliary variables $z_{ic} \to \mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)}$ approach their corresponding filter responses, and Eq. (7.4) converges to the original Eq. (7.3). This approach has been popularized for non-blind image deconvolution by Krishnan and Fergus [2009] in recent years. However, note that this continuation scheme only provides approximate HQ inference, since each assignment of the latent variables only provides a quadratic *approximation*, not a *bound* to the original penalty function (*cf.* Fig. 7.1(a,d)).

To see why this approach is so appealing, it is instructive to take a closer look at the alternating optimization procedure, which is summarized in Alg. 7.1. Specifically, the two update steps are computationally very inexpensive when – what we assume from now on –

2D convolution is carried out with circular (periodic) boundary conditions. Then we can write the two algorithmic steps as:

$$f_{i,\beta}(v) = \arg\min_{z} \left(\rho_i(z) + \frac{\beta}{2}(v-z)^2 \right) \tag{7.5}$$

$$\begin{aligned} g_\beta(\mathbf{z}) &= \left[\frac{\lambda}{\beta}\mathbf{K}^\mathsf{T}\mathbf{K} + \sum_{i=1}^{N} \mathbf{F}_i^\mathsf{T}\mathbf{F}_i \right]^{-1} \left[\frac{\lambda}{\beta}\mathbf{K}^\mathsf{T}\mathbf{y} + \sum_{i=1}^{N} \mathbf{F}_i^\mathsf{T}\mathbf{z}_i \right] \\ &= \mathcal{F}^{-1}\left[\frac{\mathcal{F}\left(\frac{\lambda}{\beta}\mathbf{K}^\mathsf{T}\mathbf{y} + \sum_{i=1}^{N} \mathbf{F}_i^\mathsf{T}\mathbf{z}_i \right)}{\frac{\lambda}{\beta}|\check{\mathbf{K}}|^2 + \sum_{i=1}^{N} |\check{\mathbf{F}}_i|^2} \right]. \end{aligned} \tag{7.6}$$

$\mathbf{F}\mathbf{x} = [\mathbf{f}^\mathsf{T}\mathbf{x}_{(\mathcal{C}_1)}, \ldots, \mathbf{f}^\mathsf{T}\mathbf{x}_{(\mathcal{C}_{|\mathcal{C}|})}]^\mathsf{T} \equiv \mathbf{f} \otimes \mathbf{x}$ denotes 2D convolution with filter $\mathbf{f}$. The optical transfer function $\check{\mathbf{F}} \equiv \mathcal{F}(\mathbf{f})$ is derived from filter (point spread function) $\mathbf{f}$, where $\mathcal{F}$ denotes the discrete Fourier transform (DFT). Note that division is applied element-wise in Eq. (7.6).

Eq. (7.5) is very cheap to compute because $f_{i,\beta}(v)$ is a univariate function that can be precomputed for all possible values of v and then stored in a lookup-table for fast retrieval [Krishnan and Fergus, 2009]. Crucially, only the additive half-quadratic form allows updating the image $\mathbf{x}$ via Eq. (7.6) very quickly in closed form, because all convolution matrices (and thus the whole equation system matrix) can be diagonalized by DFTs (*cf.* Section 3.5.1.2), which means that solving the system of linear equations amounts to element-wise division in the transformed domain followed by an inverse DFT to retain the solution in the spatial domain [*e.g.*, Krishnan and Fergus, 2009; Wang et al., 2008]. Note that this only takes $N+1$ convolutions[2] and $N+3$ DFTs with an overall complexity of $\mathcal{O}(D \log D)$, where D is the number of pixels. Note that if the size of the image is known in advance, only 2 DFTs and N convolutions are required at each iteration of Alg. 7.1, since all other major computations can be pre-computed. However, we do not make such an assumption here.

7.2.1 *Shrinkage function*

The role of $f_{i,\beta}$ (Eq. 7.5) is known as a *shrinkage* (or *mapping*) function in the wavelet image restoration literature [*cf.* Hel-Or and Shaked, 2008]. It is also known as the *proximal operator* [*cf.* Parikh and Boyd, 2013, § 1.1] of the penalty function ρ_i (with parameter β^{-1}). Intuitively, its purpose is to shrink small filter/wavelet coefficients, *i.e.* pull them towards zero, because they are assumed to be caused by noise instead of signal.

For now, the shape of the shrinkage function is determined solely by β and its associated penalty function ρ_i (Eq. 7.5, see Fig. 7.1(a–d) for an illustration). However, we make the observation that all $f_{i,\beta}$

---

[2] Each convolution can be expressed through DFTs, but typically is computationally more expensive for the small filters $\mathbf{f}_i$ used in practice.

according to Eq. (7.5) are monotonically increasing functions, regardless of the penalty ρ_i. In order to prove this Proposition 1, it is useful to first have the following Lemma:

Lemma 1. *For any function* $f : \mathbb{R} \to \mathbb{R}$ *and all* $\epsilon \geq 0$, $\arg\min_z f(z) \leq \arg\min_z (f(z) - \epsilon z)$.

Proof. If $\epsilon = 0$, Lemma 1 is trivially true; hence assume $\epsilon > 0$ from now on. Let us define the auxiliary function $g(z) = f(z) - \epsilon z$, and denote as $\hat{z}_f = \arg\min_z f(z)$ and $\hat{z}_g = \arg\min_z g(z)$ the arguments that minimize f and g, respectively. Using these definitions, it is evident that the inequalities

We thank StackExchange (Mathematics) *user* JiK *for their advice.*

$$f(\hat{z}_g) \geq f(\hat{z}_f) = \min_z f(z) \tag{7.7}$$

$$g(\hat{z}_f) \geq g(\hat{z}_g) = \min_z g(z) \tag{7.8}$$

hold. With these two inequalities, we can prove the Lemma:

$$\begin{aligned}
& g(\hat{z}_f) \geq g(\hat{z}_g) && (7.9)\\
\Rightarrow \quad & f(\hat{z}_f) - \epsilon\hat{z}_f \geq f(\hat{z}_g) - \epsilon\hat{z}_g && (7.10)\\
\Rightarrow \quad & f(\hat{z}_f) - \epsilon\hat{z}_f \geq f(\hat{z}_f) - \epsilon\hat{z}_g && (7.11)\\
\Rightarrow \quad & \hat{z}_f \leq \hat{z}_g && (7.12)\\
\Rightarrow \quad & \arg\min_z f(z) \leq \arg\min_z (f(z) - \epsilon z) && (7.13)
\end{aligned}$$

□

Proposition 1. *For all* $\epsilon, \beta \geq 0, v \in \mathbb{R}$ *and any* $\rho(z)$, *the shrinkage function* $f_\beta(v) = \arg\min_z \left(\rho(z) + \frac{\beta}{2}(v-z)^2\right)$ *is monotonically increasing,* i.e. $f_\beta(v) \leq f_\beta(v+\epsilon)$.

Proof.

$$\begin{aligned}
f_\beta(v+\epsilon) &= \arg\min_z \left(\rho(z) + \frac{\beta}{2}(v+\epsilon-z)^2\right) && (7.14)\\
&= \arg\min_z \left(\rho(z) + \frac{\beta}{2}(v-z)^2 - \epsilon\beta z\right) && (7.15)
\end{aligned}$$

It follows from Lemma 1 that $f_\beta(v) \leq f_\beta(v+\epsilon)$. □

Although shrinkage functions and proximal operators have been studied extensively in the literature, we are not aware of previous work that has observed Proposition 1 before. More importantly, it implies that one can gain additional flexibility in additive half-quadratic optimization by directly modeling the shrinkage function instead of the potential function.

As we just motivated and will further justify below, directly modeling the shrinkage function is appealing. To that end, we remove the potential function and the associated optimization problem in Eq. (7.5) altogether, and replace $f_{i,\beta}$ with a flexible shrinkage function modeled as a linear combination of Gaussian RBF kernels:

$$f_{\pi_i}(v) = \sum_{j=1}^{M} \pi_{ij} \exp\left(-\frac{\gamma}{2}(v-\mu_j)^2\right). \tag{7.16}$$

We assume shared precision γ and place the kernels at fixed, equidistant positions μ_j. We use up to $M = 53$ Gaussian kernels and make no further assumptions about the shape of the function (two examples are shown in Fig. 7.1(e–f)).

Shrinkage functions are widely studied in the wavelet restoration literature. However, instead of manually choosing shrinkage functions, we learn them from data[3] through setting the weights π_{ij} of the parametric form of Eq. (7.16). This is in clear contrast to previous work. Attempts at discriminatively learning shrinkage functions for wavelet restoration exist [*e. g.*, Hel-Or and Shaked, 2008], but are not common. Furthermore, wavelet image restoration is quite different because the pixels of the restored image are not connected via a random field, as here.

We are not aware of any previous work that has used learning in the context of this particular form of half-quadratic optimization. Consequently, the full potential of this fast optimization approach has not been unlocked, because model parameters have always been chosen by hand. Furthermore, the β-continuation schedule for the number of iterations of Alg. 7.1 is typically manually chosen.

In the following, we show how to overcome all of these limitations while retaining the computational benefits of this approach. To that end, we learn all model parameters (other than the size and number of filters, and the number of optimization iterations) from training data.

The most important benefit of directly modeling the shrinkage functions is that it allows us to reduce the optimization procedure to a single quadratic minimization in each iteration, which we denote as the prediction of a *shrinkage field* (SF):

$$\begin{aligned} g_{\boldsymbol{\Theta}}(\mathbf{x}) &= \mathcal{F}^{-1}\left[\frac{\mathcal{F}\left(\lambda \mathbf{K}^{\mathsf{T}}\mathbf{y} + \sum_{i=1}^{N} \mathbf{F}_i^{\mathsf{T}} f_{\pi_i}(\mathbf{F}_i\mathbf{x})\right)}{\lambda|\check{\mathbf{K}}|^2 + \sum_{i=1}^{N}|\check{\mathbf{F}}_i|^2}\right] && (7.17) \\ &= \boldsymbol{\Omega}^{-1}\boldsymbol{\eta}. && (7.18) \end{aligned}$$

---

[3] A possibly more suitable name would be *mapping* instead of *shrinkage* function, since our learned functions do not necessarily shrink the associated filter responses. We keep the widely known name despite this.

Algorithm 7.2 Inference with a cascade of shrinkage fields

```
$\hat{\mathbf{x}}_0 \leftarrow \mathbf{y}$
for $t \leftarrow 1$ to $T$ do
    $\hat{\mathbf{x}}_t \leftarrow g_{\boldsymbol{\Theta}_t}(\hat{\mathbf{x}}_{t-1})$
```

A shrinkage field $\mathcal{N}(\boldsymbol{\Omega}^{-1}\boldsymbol{\eta}, \boldsymbol{\Omega}^{-1})$ is thus a particular Gaussian conditional random field, whose moments $\boldsymbol{\eta}$ and $\boldsymbol{\Omega}$ are determined through learned model parameters $\boldsymbol{\Theta}$, the observed image $\mathbf{y}$, and the point spread function $\mathbf{k}$. A key benefit is that the shrinkage field prediction $g_{\boldsymbol{\Theta}}(\mathbf{x})$ and its gradient $\frac{\partial g_{\boldsymbol{\Theta}}(\mathbf{x})}{\partial \boldsymbol{\Theta}}$ w.r.t. the model parameters $\boldsymbol{\Theta}$ can be computed in closed form, which allows for efficient parameter learning (Section 7.3.1). This is in contrast to more complicated learning procedures in other formulations, which need to solve nested minimization problems using bi-level optimization (Section 2.4.1, [*e.g.*, Samuel and Tappen, 2009; Chen et al., 2013]). Note that we completely eliminate the continuation parameter β, which is absorbed into the weights $\boldsymbol{\pi}_i$ of Eq. (7.16) and fused with λ (which will be learned) in Eq. (7.17).

Since half-quadratic optimization typically involves several (many) iterations of Eqs. (7.5) and (7.6), we can similarly chain multiple predictions into a *cascade of shrinkage fields* (CSF), as summarized in Alg. 7.2. A CSF is thus a cascade of Gaussian CRFs (Chapter 6). Note that the concept of a shrinkage function does not exist in previous CRF cascades. RTF cascades (Chapter 6), for example, use regression trees to specify unary and pairwise factors; since the resulting equation system matrix cannot be diagonalized by DFTs, they *do not* admit fast closed-form inference as in Eq. (7.17).

7.3.1 *Learning*

We learn the model parameters $\boldsymbol{\Theta}_t = \{\lambda_t, \boldsymbol{\pi}_{ti}, \mathbf{f}_{ti}\}_{i=1}^N$ through loss-minimization for every stage (iteration) t of Alg. 7.2. By learning different model parameters for every stage of our cascade, we essentially learn tailored random field models for each iteration of the associated optimization algorithm[4]. For non-blind deconvolution, we follow Chapter 6 and parameterize the prediction with the blur kernel, such that the instance-specific blur ($\mathbf{K}$ in Eq. 7.17) is provided at test time; models are *not* trained for specific blurs.

To greedily learn the model stage-by-stage from $t = 1, \ldots, T$, at stage t we minimize the cost function

$$J(\boldsymbol{\Theta}_t) = \sum_{s=1}^{S} \ell(\hat{\mathbf{x}}_t^{(s)}; \mathbf{x}_{\text{gt}}^{(s)}) \qquad (7.19)$$

---

[4] However, if we used the same filters at each model stage, we could re-use all optical transfer functions and save a lot of runtime after stage 1.

with training data $\{\mathbf{x}_{\text{gt}}^{(s)}, \mathbf{y}^{(s)}, \mathbf{k}^{(s)}\}_{s=1}^{S}$, where $\hat{\mathbf{x}}_t^{(s)}$ is obtained with Alg. 7.2. We can, in principle, employ any continuously differentiable loss function, and concretely choose the (negative) *peak signal-to-noise ratio* (PSNR)

$$\ell(\hat{\mathbf{x}}; \mathbf{x}_{\text{gt}}) = -20 \log_{10} \left(\frac{R\sqrt{D}}{\|\hat{\mathbf{x}} - \mathbf{x}_{\text{gt}}\|} \right), \tag{7.20}$$

where D denotes the number of pixels of $\hat{\mathbf{x}}$ and R the maximum intensity level of a pixel (*i. e.*, $R = 255$).

We minimize Eq. (7.19) with the gradient-based L-BFGS method (using an implementation by Schmidt [2005]). To that end, we, akin to Jancsary et al. [2012a], differentiate the loss of the predicted restored image $\hat{\mathbf{x}}_t$ (at stage t) w.r.t. model parameters $\boldsymbol{\Theta}_t$ as

$$\begin{aligned} \frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\text{gt}})}{\partial \boldsymbol{\Theta}_t} &= \frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\text{gt}})}{\partial \hat{\mathbf{x}}_t} \cdot \frac{\partial \boldsymbol{\Omega}_t^{-1} \boldsymbol{\eta}_t}{\partial \boldsymbol{\Theta}_t} && (7.21) \\ &= \hat{\mathbf{c}}_t^{\mathsf{T}} \left[\frac{\partial \boldsymbol{\eta}_t}{\partial \boldsymbol{\Theta}_t} - \frac{\partial \boldsymbol{\Omega}_t}{\partial \boldsymbol{\Theta}_t} \hat{\mathbf{x}}_t \right] && (7.22) \\ \text{with } \hat{\mathbf{c}}_t &= \boldsymbol{\Omega}_t^{-1} \left[\frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\text{gt}})}{\partial \hat{\mathbf{x}}_t} \right]^{\mathsf{T}}. \end{aligned}$$

Similar to $\hat{\mathbf{x}}_t$, we can efficiently compute $\hat{\mathbf{c}}_t$ by solving a system of linear equations via element-wise division in the transformed domain. The derivatives for specific model parameters as well as further details, such as boundary handling due to periodic convolutions and parameter constraints, are omitted here for brevity and to make the equations more readable; however, all details can be found in Appendix A.2.

In Eq. (7.19), each stage is trained greedily such that the loss is as small as possible after each stage, regardless of how many stages T are actually intended to be used in the cascade; this also applies to the cascade model of Chapter 6. However, in contrast to the cascade of Chapter 6, which uses non-differentiable regression trees to determine the parameters of a Gaussian CRF and requires custom training, our shrinkage functions are smooth and differentiable. Hence, we do not need to alternate between gradient-based and combinatorial optimization (growing regression trees). Moreover, we can use standard gradient-based methods to jointly train all T stages of the model by minimizing

$$J(\boldsymbol{\Theta}_{1,\dots,T}) = \sum_{s=1}^{S} \ell(\hat{\mathbf{x}}_T^{(s)}; \mathbf{x}_{\text{gt}}^{(s)}), \tag{7.23}$$

where only the loss of the final prediction $\hat{\mathbf{x}}_T$ is relevant. The derivatives w.r.t. model parameters of all stages can be computed efficiently and take the same basic form as Eq. (7.22), which allows for an easy implementation. This bears similarities to deep (convolutional) neural networks, with the difference that in our case all nodes of a layer are fully-connected due to the DFT-based inference step. Note that

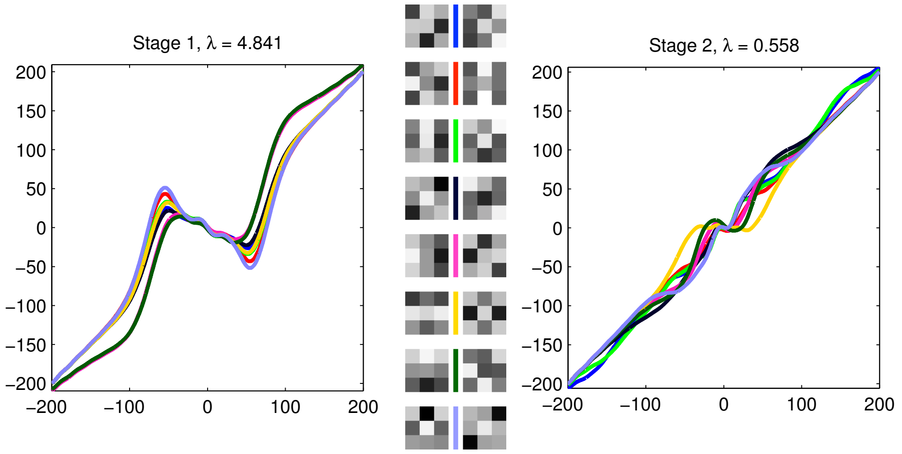


Figure 7.2: **First two stages of learned CSF$_{3\times3}$ model.** The shrinkage functions are color-matched with their corresponding filters.

all stages can be learned jointly even while applying boundary operations, such as padding and truncation. All details and derivations are in Appendix A.2.

7.4 EXPERIMENTS

TRAINING Although the class of Gaussian CRFs that can be learned at one stage of our approach is restricted (compared to Jancsary et al. [2012a]), this limitation comes at the substantial benefit of fast prediction and learning. That means we can train our model on relatively large datasets – even with a simple MATLAB implementation[5]. To generate the training data for our denoising experiments, we cropped a 256×256 pixel region from each of 400 images of the Berkeley segmentation dataset [Martin et al., 2001][6], *i. e.* our training set thus roughly contains 25 million pixels.

We have greedily trained 5 stages of four different configurations of our model with increasing capacity:

CSF$^{5}_{\text{PW.}}$ Pairwise model with fixed $\mathbf{f} = \left\{[1,-1],[1,-1]^{\mathsf{T}}\right\}$.

CSF$^{5}_{3\times3}$ Fully trained model with 8 filters of size 3×3.

CSF$^{5}_{5\times5}$ Fully trained model with 24 filters of size 5×5.

CSF$^{5}_{7\times7}$ Fully trained model with 48 filters of size 7×7.

Hence, CSF$^{T}_{m\times m}$ denotes a cascade of T stages with $m^2 - 1$ filters of size $m\times m$ (if $T < 5$, only T stages have been evaluated at test time; prediction can be stopped at any stage). Note that many more configurations are possible and will lead to different performance vs. speed

---

[5] Code for learning and inference is available on the author's webpage.

[6] These are strictly separate from all test images in Tabs. 7.2 and 7.1.

Method	PSNR	St.	$\text{CSF}_{\text{pw.}}$	$\text{CSF}_{3\times3}$	$\text{CSF}_{5\times5}$	$\text{CSF}_{7\times7}$
BLS-GSM [Portilla et al., 2003]	27.98	1	26.60	27.54	27.46	27.70
5×5 FoE [Gao and Roth, 2012]	28.22	2	27.26	27.93	28.26	28.38
LSSC [Mairal et al., 2009]	28.23	3	27.31	28.02	28.34	28.45
BM3D [Dabov et al., 2007b]	28.31	4	27.36	28.05	28.37	28.52
RTF5 [Schmidt et al., 2016]	**28.74**	5	27.36	28.08	28.39	**28.53**

Table 7.1: Average PSNR (dB) on 68 images from [Roth and Black, 2009] for image denoising with $\sigma = 25$. On the right, each row shows the results from the respective stage of our models.

tradeoffs, which can be chosen to suit the particular application. The first two stages of the learned $\text{CSF}_{3\times3}$ and $\text{CSF}_{\text{pw.}}$ models are shown in Figs. 7.2 and 7.1(e–f), respectively, which are good examples of our observation that almost all learned shrinkage functions are *not* monotonically increasing, which means they could not have been obtained by learning a potential function (*cf.* Section 7.2).

DENOISING We first evaluated the task of image denoising (*i.e.*, $\mathbf{k} = 1$), for which we trained our models to remove Gaussian noise with standard deviation $\sigma = 25$. The noisy training images were obtained by adding simulated Gaussian noise to the clean images. We subsequently quantized the intensity values of the noisy images to 8-bit to make the training data somewhat more realistic. In practice, noisy images are always integer-valued and range-limited, such as intensity values being in $\{0, \ldots, 255\}$.

After training the models, we evaluate them on 68 (8-bit quantized noisy) test images originally introduced by Roth and Black [2009], which have since become a reference set for image denoising; Fig. 7.4 shows a denoising example. We compare against a varied selection of recent state-of-the-art techniques. The results in Tab. 7.1 show that the (5-stage) cascade of regression tree fields (RTFs) from Chapter 6 achieves the best performance (trained with the same data as our CSF models here). This is not surprising, since the more flexible RTFs do not make any noise assumption (in contrast to all other approaches in Tab. 7.1) and can thus effectively handle the additional quantization noise. Concerning the other methods, we outperform the strongest competitor BM3D by 0.22dB with our most powerful $\text{CSF}^{5}_{7\times7}$ model. Furthermore, our $\text{CSF}^{4}_{5\times5}$ model slightly outperforms BM3D and also has a faster runtime (*cf.* Fig. 7.3), even when only the CPU is used. Additionally, our model's inference procedure (convolutions and DFTs being the most expensive operations) is presumably much more amenable to GPU or DSP parallelization than the block-matching procedure of BM3D. It can also be observed that results of our models saturate after only 3–4 stages, hence "converge" very quickly.

Method	σ=15	σ=25
KSVD [Elad and Aharon, 2006]	30.87	28.28
FoE [Gao and Roth, 2012]	30.99	28.40
BM3D [Dabov et al., 2007b]	31.08	28.56
opt-MRF [Chen et al., 2013]	31.18	28.66
EPLL [Zoran and Weiss, 2011]	31.19	28.68
LSSC [Mairal et al., 2009]	**31.27**	**28.70**
ARF-4 [Barbu, 2009]	30.70	28.20
RTF$_5$ [Schmidt et al., 2016]	—	**28.75**
$\text{CSF}^{5}_{\text{pw.}}$	29.99	27.47
$\text{CSF}^{4}_{3\times3}$	30.78	28.29
$\text{CSF}^{4}_{5\times5}$	31.12	28.58
$\text{CSF}^{5}_{7\times7}$	**31.24**	**28.72**

Table 7.2: Average PSNR (dB) on 68 images from [Roth and Black, 2009] for image denoising with $\sigma = 15, 25$; top part quoted from [Chen et al., 2013]. Training of our CSF models and denoising carried out *without* 8-bit quantization of noisy images to allow comparison with [Barbu, 2009] and [Chen et al., 2013].

We also compare against the recently introduced *opt-MRF* by Chen et al. [2013] for two reasons: First, it currently is one of the best-performing CRFs for image restoration, achieved by using better optimization techniques with a model architecture originally proposed by Samuel and Tappen [2009]. Secondly, it uses a model configuration very similar to ours, that is 48 filters of size 7×7, which are fully learned from data (including associated potential functions). Moreover, we compare against the fast *active random field* (ARF) model of Barbu [2009], which uses 24 filters of size 5×5. Since both of them were neither trained nor evaluated with 8-bit quantized noisy images, we use their setting to not give our model an unfair advantage. Hence, we additionally trained and evaluated our models without quantization. The results in Tab. 7.2 show[7] that we outperform [Barbu, 2009; Chen et al., 2013], and can also compete with the RTF-based cascade model (trained with non-quantized images, *cf.* Chapter 6), whose additional flexibility does not seem pay off here since the image noise is truly Gaussian. The results further show that we can also compete for noise level $\sigma = 15$, for which we trained additional models.

---

[7] Comparing Tabs. 7.2 and 7.1 also shows how much results improve when they are obtained in a more artificial (non-quantized) setting.

RUNTIME The runtime comparison[8] for image denoising in Fig. 7.3 shows that our model scales to image sizes of more than 16 megapixels at reasonable runtimes (at most 10 minutes for our best model with a simple single-threaded MATLAB implementation, and only 23 seconds on a GPU).

While a cascade of RTFs (Chapter 6) is very flexible and yields state-of-the-art restoration results, its relatively complex and highly optimized C++ implementation hinges on multi-threading to boost runtime performance. Comparing single-threaded performance (Fig. 7.3), it is about an order of magnitude slower compared to our $\text{CSF}_{7\times7}$ (which exhibits competitive performance, *cf.* Tab. 7.2). We outperform BM3D at a faster runtime with our $\text{CSF}^4_{5\times5}$ model (*cf.* Tab. 7.1).

Additionally, our model's inference procedure is well suited for GPU or DSP parallelization. In order to gauge the potential speedup, we used the same code with MATLAB's built-in GPU capabilities and were able to obtain significantly improved runtimes (Tab. 7.3). However, we should expect additional speedups by using a more powerful recent GPU with an optimized implementation using CUDA or OpenCL (Chen et al. [2013] quote a 40× GPU speedup over presumably multi-threaded CPU code).

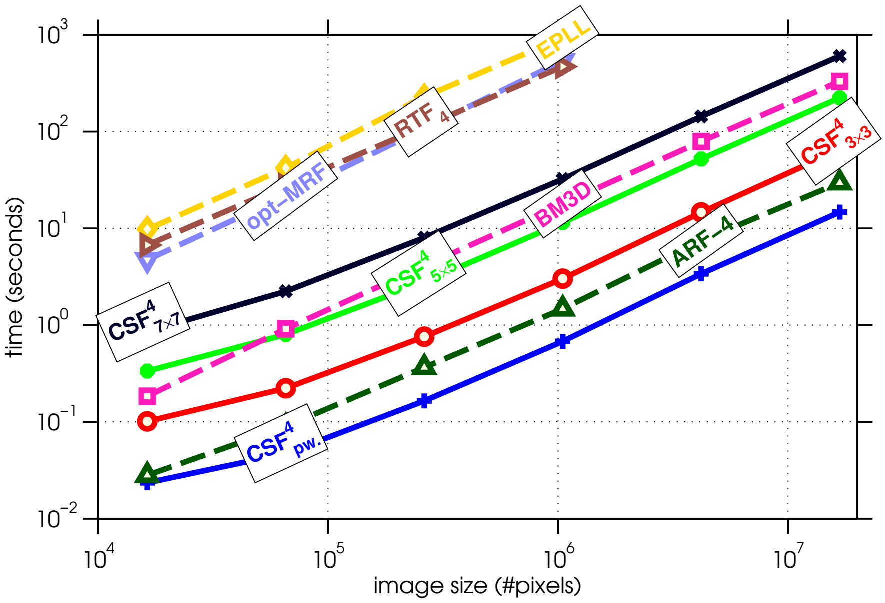
While the ARF model [Barbu, 2009] (designed for real-time denoising) is more efficient (CPU only) than our CSF with the same number of stages, filters, and filter size, it exhibits inferior results: It performs 0.38dB worse than $\text{CSF}^4_{5\times5}$ (Tab. 7.2), and even our $\text{CSF}^4_{3\times3}$ model with only 8 3×3 filters surpasses the ARF in terms of restoration quality. While the ARF is twice as fast as $\text{CSF}^4_{3\times3}$, we can speed CSF up by re-using filters (*cf.* Section 7.3.1). Furthermore, our standard gradient-based learning procedure is much easier and faster, and enables learning more powerful models such as $\text{CSF}^5_{7\times7}$.

Computing the learning objective function $J(\boldsymbol{\Theta})$ (Eq. 7.19) and its gradient $\partial J(\boldsymbol{\Theta})/\partial\boldsymbol{\Theta}$ for $S = 400$ images of 256×256 pixels takes in total only around 7s ($\text{CSF}_{\text{pw.}}$), 24s ($\text{CSF}_{3\times3}$), 73s ($\text{CSF}_{5\times5}$), or 161s ($\text{CSF}_{7\times7}$) with our simple MATLAB implementation (Intel Core i7-3930K hexa-core at 3.20GHz, six parallel threads). This allows us to thoroughly train our models by using 200 L-BFGS iterations. Another important property of our method is its predictable runtime, which is in contrast to methods (such as opt-MRF and RTF) that require iterative inference whose convergence depends on the input data. In our experience, runtime varies even more for deconvolution, mostly due to the blur kernel.

JOINT TRAINING While jointly training all stages of the model has the potential to yield superior results, we only partly confirm this in our denoising experiments. Since our learning objective function is

---

[8] MATLAB/C++ implementations from the respective authors, single-threading strictly enforced (including `-singleCompThread` option for MATLAB).



Method	128^2	256^2	512^2	1024^2	2048^2	4096^2
$\text{CSF}^4_{\text{pw.}}$	0.02	0.05	0.17 (0.03)	0.7 (0.05)	3.4 (0.18)	15 (0.8)
$\text{CSF}^4_{3\times3}$	0.10	0.22	0.76 (0.15)	3.0 (0.27)	14.6 (0.78)	65 (3.7)
$\text{CSF}^4_{5\times5}$	0.34	0.80	2.78 (0.44)	11.5 (0.80)	52.0 (2.42)	223 (10.8)
$\text{CSF}^4_{7\times7}$	0.86	2.23	8.00 (0.92)	32.3 (1.72)	143 (5.27)	603 (23.2)
ARF-4 [Barbu, 2009]	0.03	0.09	0.37	1.5	7.5	29
BM3D [Dabov et al., 2007b]	0.18	0.92	4.09	18.0	78.9	330
opt-MRF [Chen et al., 2013]	4.73	21.7	108	538	–	–
RTF$_4$ [Schmidt et al., 2016]	6.71	27.7	113	469	–	–
EPLL [Zoran and Weiss, 2011]	9.76	41.9	229	930	–	–

Figure 7.3: **Runtime comparison for image denoising.** Single-threaded runtimes (in seconds) with an Intel Core i7-3930K CPU at 3.20GHz; small numbers in parentheses from simple MATLAB-based GPU execution on a NVIDIA GeForce GTX 480. Runtimes of our models shown after 4 stages where performance saturates; using fewer stages takes proportionally less time, *e.g.* 2 stages take half the time. Note the logarithmic scales on both axes *(top)*. The table columns show runtimes for image sizes up to 4096×4096 pixels (about 16.78 megapixels).

not convex, the optimization often gets stuck in worse local optima than when using greedy training. Hence we tried first training each stage greedily (pre-training), and then "tuned" the model by starting joint training with the parameters obtained from pre-training. While this is guaranteed to not decrease (training set) performance, it does not always improve results considerably, especially with increasing model capacity. Jointly tuning all 5 stages of $\text{CSF}^5_{\text{pw.}}$ does pay off, by increasing PSNR performance about 0.31dB from 27.36dB to 27.67dB

Blur kernel	Levin et al.	Schmidt et al.	$\text{CSF}^1_{\text{pw.}}$	$\text{CSF}^2_{\text{pw.}}$	$\text{CSF}^3_{\text{pw.}}$
Ground truth	32.73	**33.97**	32.48	33.50	33.48
Levin et al. [2011]	30.05	**30.40**	29.63	30.34	**30.42**
Cho and Lee [2009]	29.71	29.73	29.10	29.86	**29.99**
Fergus et al. [2006]	28.38	**29.10**	28.36	29.02	29.01

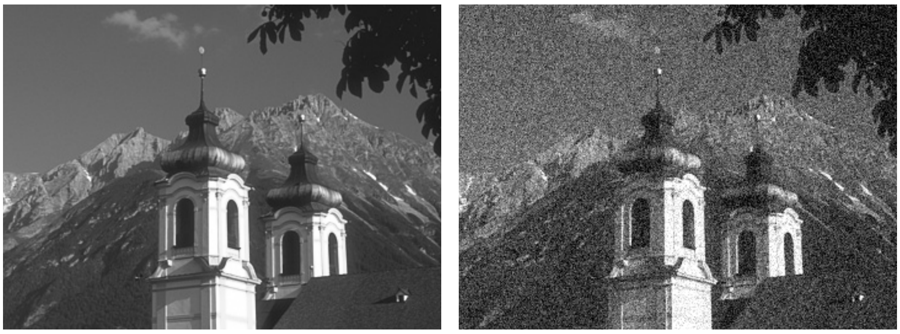
Table 7.3: Average PSNR (dB) on 32 images from [Levin et al., 2011] for image deconvolution. Rows correspond to different blur kernel (estimates) provided by [Levin et al., 2011], while columns correspond to non-blind deconvolution methods. Left part of table reproduced from Chapter 6, showing results from [Levin et al., 2007] and [Schmidt et al., 2016].

(*cf*. Tab. 7.1). However, tuning all 5 stages of our other models hardly makes a difference. Even for 3-stage tuning we observe only minor improvements, *e.g.* from 28.02dB to 28.09dB for $\text{CSF}^3_{3\times3}$, and from 28.34dB to 28.36dB for $\text{CSF}^3_{5\times5}$.

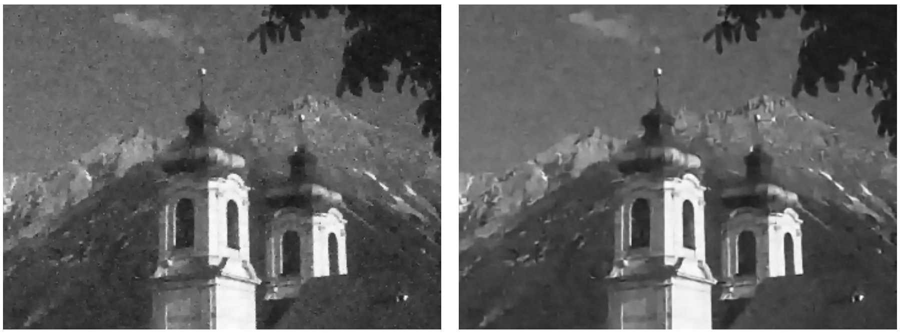
NON-BLIND DECONVOLUTION As the results in Tab. 7.3 show, our approach can also successfully be applied to image deconvolution in the context of blind deblurring, where kernel estimates are used to deblur the image. For the task of deconvolution, we trained a $\text{CSF}_{\text{pw.}}$ model with 288 synthetically blurred images of size 320×320 pixels. For half of the blurred training images, we used an estimate instead of the correct blur kernel $\mathbf{k}$ to cope with using erroneous kernel estimates at test time (as we did in Chapter 6). Our $\text{CSF}^3_{\text{pw.}}$ model outperforms the non-blind deconvolution approach by Levin et al. [2007] and can compete with the results from Chapter 6 [Schmidt et al., 2016] for all estimated kernels (Tab. 7.3); see Fig. 7.5 for a qualitative comparison. We additionally applied the same learned $\text{CSF}^3_{\text{pw.}}$ model to the recent benchmark for camera shake of Köhler et al. [2012], where we are able to improve upon the results of the best performing method by Xu and Jia [2010] about 0.56dB on average, being also 0.15dB better than the best result of Chapter 6. Restoring each of the 800×800-sized color images of the benchmark only takes around a second with our model.

7.5 SUMMARY

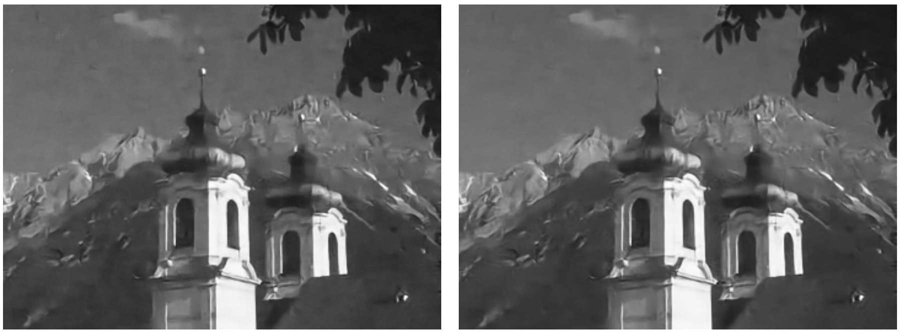
We presented shrinkage fields, a novel random field model applicable to the restoration of high-resolution images. As in Chapter 6, our approach is based on a generalization of half-quadratic optimization. However, in contrast to the previous chapter we specifically extended the additive HQ form, which admits very fast inference with predictable runtime. By replacing potentials with shrinkage functions, we increased model flexibility and enabled efficient end-to-end learning of all model parameters with standard gradient-based methods. Experiments on image denoising and deconvolution with cascaded shrinkage fields demonstrated that fast runtime and high restoration quality can go hand-in-hand.



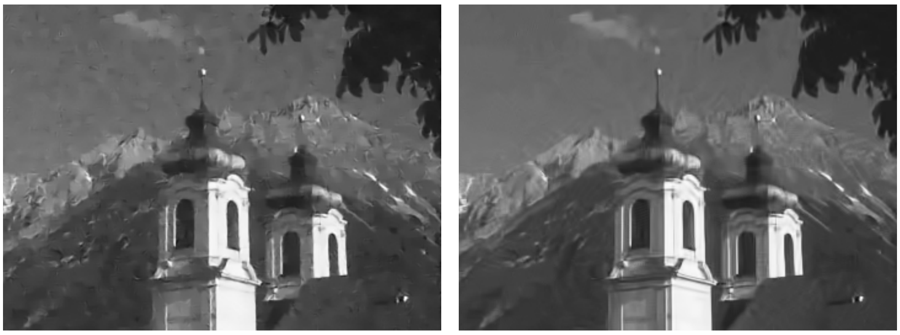
(a) Original image (b) Noisy, 20.30dB



(c) csf$^{5}_{\text{pw.}}$, 28.81dB (d) csf$^{5}_{3\times3}$, 29.89dB

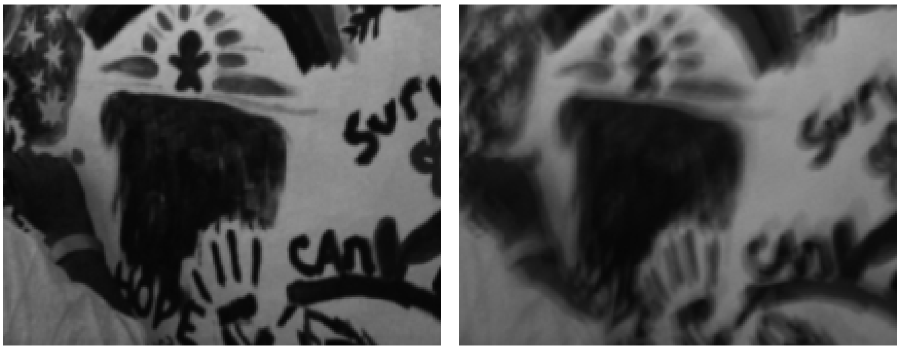


(e) csf$^{5}_{5\times5}$, 30.27dB (f) csf$^{5}_{7\times7}$, 30.38dB

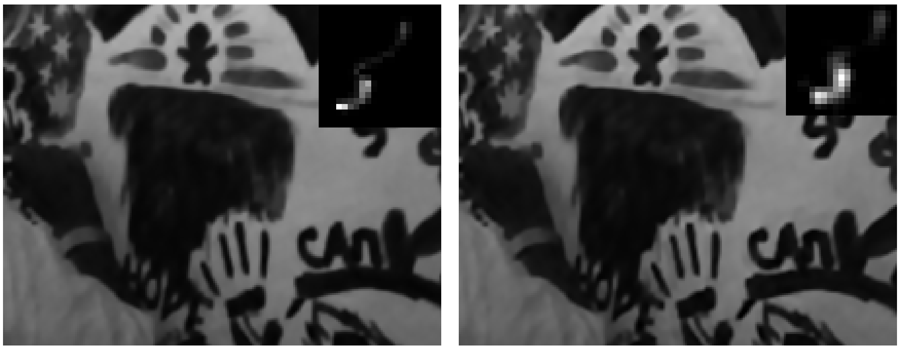


(g) ARF-4, 29.76dB (h) BM3D, 30.05dB

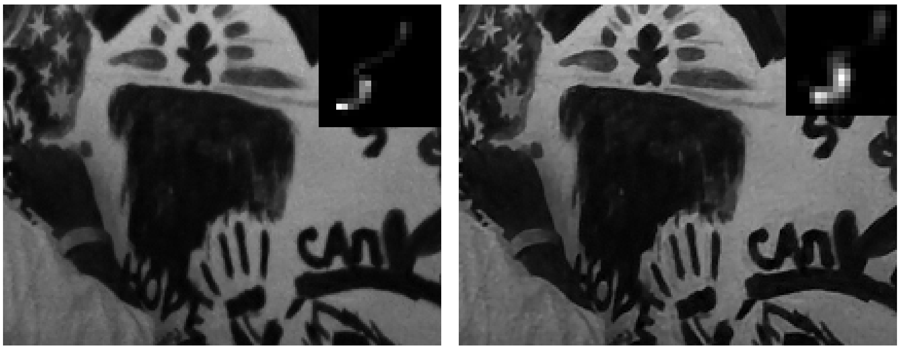
Figure 7.4: **Denoising example** ($\sigma = 25$, cropped): Comparison of our trained models with BM3D [Dabov et al., 2007b] and ARF [Barbu, 2009]. *Best viewed magnified on screen.*



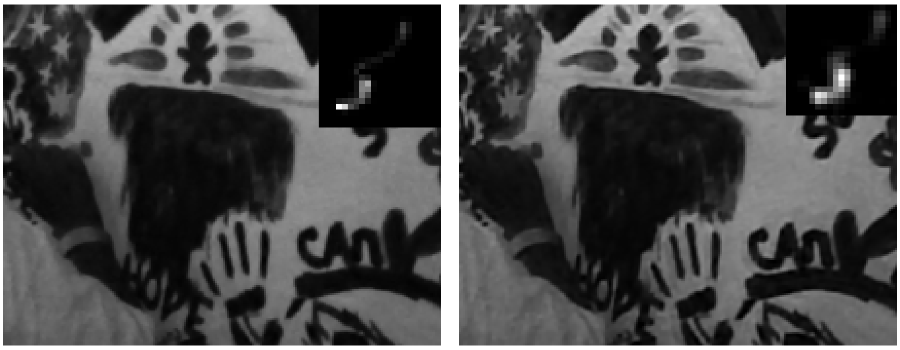
(a) Original image (b) Blurred, 23.78dB



(c) Levin et al. [2007], 37.03dB (d) Levin et al. [2007], 33.79dB



(e) RTF (Chapter 6), 38.04dB (f) RTF (Chapter 6), 33.90dB



(g) $\text{CSF}^{3}_{\text{pw.}}$, 38.02dB (h) $\text{CSF}^{3}_{\text{pw.}}$, 34.16dB

Figure 7.5: **Deconvolution example** (cropped): Comparison for image deconvolution (*cf.* Table 7.3) with different blur kernels (ground truth: *(c,e,g)*; estimate via [Levin et al., 2011]: *(d,f,h)*). *Best viewed magnified on screen.*

8

SUMMARY AND FUTURE WORK

CONTENTS

8.1 Contributions 175
  8.1.1 Generative models 175
  8.1.2 Discriminative models 177
8.2 Discussion and Outlook 178
  8.2.1 Generative and discriminative approaches 178
  8.2.2 Half-quadratic sampling 179
  8.2.3 Jointly learning model and inference 180
  8.2.4 Towards richer and more accurate models 182

SUITABLE random field models for natural images necessitate the use of edge-preserving potential functions, which in turn lead to challenging inference and learning problems, especially in a generative context (*cf.* Chapter 2). To alleviate these issues, we have employed and extended half-quadratic (HQ) techniques throughout this dissertation, because they are effective at converting challenging optimization problems into a sequence of easier ones (*cf.* Chapter 3).

We considered applications in generative and discriminative contexts, because both strategies have their advantages and disadvantages (*cf.* Section 2.5). In this final chapter, we not only summarize our contributions but also discuss limitations, which often indicate promising avenues for future work.

8.1 CONTRIBUTIONS

8.1.1 *Generative models*

LIKELIHOOD MODEL WITH UNKNOWN PARAMETERS In a generative approach, we specify the forward (or observation) model with a likelihood, which encodes our assumption how the observed image relates to the unknown image that we want to estimate. However, the likelihood often hinges on a few crucial instance-specific parameters to be accurate (*e.g.*, the variance of assumed Gaussian noise). In Chapter 4, we addressed the issue that some of these parameters are often unknown in practice. To that end, we extended a sampling-based inference approach based on a HQ construction, which had previously been used for denoising [Schmidt et al., 2010] and deblurring [Schmidt et al., 2011]. We proposed to jointly estimate the restored image and the unknown likelihood parameters by treating the latter as additional latent variables, which we included in a Gibbs sampling framework. To that end, we obtained samples from the joint distribution of the restored image and all latent variables.

In particular, this allowed us to perform image denoising and deblurring with integrated (Gaussian) noise estimation. Furthermore, we additionally considered parametric blur estimation besides estimating the noise. Concretely, we demonstrated promising blind deconvolution results in two cases, namely under the assumptions of Gaussian blur and linear (camera) motion blur. Our approach is conceptually very appealing, since it only relies on a likelihood assumption and an accurate image prior. There is no need for separate parameter tuning or estimation steps.

PRIOR MODEL WITH EXPLICIT INVARIANCES While Chapter 4 was concerned with handling unknown parameters of the *likelihood* model, Chapter 5 focused on incorporating domain knowledge into the *prior* model. To that end, we proposed a framework for transformation-invariant product models, which distinguishes between (typically learned) linear features of the data and a set of (known) linear transformations. An important property of our approach is that it allows transformation-aware feature learning, where learned features have to be "useful" at all specified transformations; this is because this may also be thought of as implicitly adding transformed copies of each feature to the model. As a consequence, many fewer features need to be learned since implicitly added features share parameters.

Commonly-used convolutional models (*e.g.*, filter-based MRFs such as the FoE model) can be expressed in our framework when considering translation-invariance. We went beyond translations and additionally imposed (approximate) invariance to image rotations, which is often desirable but rarely modeled explicitly. Concretely, we learned a translation- and (90°) rotation-invariant FoE image prior and demonstrated its merits for rotation-equivariant image denoising. Furthermore, we extended convolutional Restricted Boltzmann Machines (RBMs) to also be invariant to rotations in 45° increments. However, we were mainly not interested in learning a translation- and rotation-invariant RBM model, but rather used the obtained features for (rotation-invariant) object recognition and detection. In particular, we exploited the known relationship between features to devise a rotation-equivariant image descriptor, which we further extended to be rotation-invariant. We showed the efficacy of our descriptor for handwritten digit recognition and car detection in satellite images.

HALF-QUADRATIC INFERENCE It is important to note that auxiliary-variable block Gibbs sampling (*cf.* Section 2.3.1.1) was an important inference component in Chapter 4 and formed the backbone for inference and learning in Chapter 5; this especially includes HQ techniques for the FoE and RBM models of natural images. While our employed half-quadratic Gibbs sampler is typically quite computationally expensive (see Section 8.2.2 below), it is a key component to enable accurate learning and inference with FoE-like image priors.

While HQ inference played an important role in enabling our contributions in a generative setting, we did not extend previous HQ methods. In contrast, our contributions in a discriminative context (see below) are directly based on HQ ideas and extensions of previous HQ approaches. Hence, we provided an extensive and unifying review of HQ inference in Chapter 3.

8.1.2 *Discriminative models*

GENERALIZATION OF HALF-QUADRATIC INFERENCE Beginning with Chapter 3, we characterized HQ inference for MAP estimation as a sequence of predictions from Gaussian MRFs, each specified through the auxiliary variables from the HQ augmentation, which are updated during the iterative inference procedure.

Based on the realization that the final prediction in half-quadratic MAP estimation comes from a Gaussian MRF (adapted to the observed image via the auxiliary variables), we discussed in Chapter 6 that one could in principle achieve the same or a similar result with a Gaussian CRF that has access to the observed image and thus may directly adapt to it. However, since it can be difficult to devise such a CRF based on (features from) the observed image, we proposed a discriminative generalization of half-quadratic MAP estimation in form of a cascade of Gaussian CRFs. While common HQ inference is retained as a special case, we can use arbitrary regression functions to determine the parameters of the Gaussian CRF at each stage of the cascade; this is in contrast to standard HQ inference which relies on fixed update equations for the auxiliary variables.

DISCRIMINATIVE NON-BLIND DEBLURRING While Gaussian CRFs had previously been used for image denoising, they had not been applied in a cascade. However, we argued and demonstrated that a cascade is crucial for more difficult problems, such as image deblurring. In particular, we proposed the first discriminative approach to non-blind image deblurring that could be applied to arbitrary (natural) images and blurs. While the cascade played an important role in obtaining state-of-the-art results, the other important component was to adapt to instance-specific blurs by integrating the blur formation assumption into the Gaussian CRF. Without this latter part, it is much more difficult to train a discriminative model that is able to work well for arbitrary blurs at test time.

CASCADE OF GAUSSIAN CRFS Specifically, we chose flexible Gaussian CRFs in the form of regression tree fields (RTFs) [Jancsary et al., 2012b,a]. We discriminatively trained a cascade of RTFs through loss minimization, which achieved excellent results for non-blind deconvolution in the context of artificially blurred and real images with blur. We also demonstrated that a cascade model can improve restoration performance for the simpler problems of grayscale and color image denoising, where we also achieve state-of-the-art performance.

DEEP SHRINKAGE FIELDS Although the cascade of RTFs admits relatively efficient (iterative) inference via a CG-based equation system solver, it does not scale very well to large mega-pixel sized images, especially for image deblurring, which often requires many iterations of CG to converge. We addressed this issue in Chapter 7 and proposed an image restoration method that scales to large images and additionally admits simplified learning of model parameters. Our approach is also motivated as an extension of half-quadratic MAP estimation. However, in contrast to Chapter 6, we extend a very specific efficient HQ variant of the additive form, which gains its efficiency by allowing to quickly solve the necessary equation systems via DFTs. Furthermore, we replace the update step of the latent variables with a parametric (shrinkage) function, which allows us to obtain the prediction of the restored image and its gradient w.r.t. all model parameters in closed form. This in turn enables discriminative learning via loss minimization with standard gradient-based methods. Although our proposed shrinkage field model, which is also a Gaussian CRF, is restricted compared to a regression tree field, we show that we can achieve similar results (for image denoising and deconvolution) with a cascade of shrinkage fields as compared to a cascade of RTFs. Additionally, shrinkage fields admit much simpler learning and inference algorithms, which can scale to large mega-pixel images as produced by modern consumer cameras.

8.2 DISCUSSION AND OUTLOOK

8.2.1 *Generative and discriminative approaches*

Generative approaches are very versatile and offer important benefits, such as a principled way to handle unobserved random variables, or assessing the “uncertainty” of estimates. We have not talked about the latter, but such estimates can be useful if one needs to know how much a prediction can be trusted, especially if it is an intermediate result in a bigger system. For example, the sampling-based inference employed in Chapter 4 also gives us an approximation of the marginal distributions of all variables, including the pixels of the restored images. The uncertainty of the estimated image could then be quantified via the entropy or variance of these marginals.

On the other hand, inference and learning with generative models are often very difficult as compared to discriminative and especially non-probabilistic approaches (*cf.* Section 2.5). In the future, it would be interesting to further combine the benefits of both approaches. For example, we did such a combination by integrating the blur likelihood into a Gaussian CRF (Chapters 6 and 7) to make it more versatile by being able to cope with varying blurs after the model has been trained. An example of a more intricate combination of generative and discriminative models has recently been proposed by Sohl-Dickstein et al. [2015], who define a flexible probability distribution through a sequence of transformations starting from a known and tractable distribution; each transformation step is carried out via discriminatively-trained models.

8.2.2 *Half-quadratic sampling*

We already mentioned several times that inference and learning of MRFs in a generative context is often computationally expensive. While HQ inference alleviates this to some degree, computation still is a problem, especially in the multiplicative form (*cf.* Chapter 3). While MAP estimation can often be carried out with only relatively few HQ iterations, this issue is more severe for sampling-based inference as we have used in Chapter 4, which often requires on the order of hundreds of Gibbs sampling iterations to yield high-quality estimates.

In Chapter 4, the culprit is repeatedly solving the equation systems that arise in the multiplicative HQ form. As discussed in Section 3.5, using a Cholesky decomposition does not scale to larger images. Furthermore, the equation system matrix is typically not well-conditioned, which means that iterative solvers, such as CG, require many iterations to converge (in our experience on the order of several thousands to obtain a small error). Although each iteration of CG only requires multiplication with the equation system matrix, which can be carried out via 2D convolutions and element-wise operations, it is overall too expensive, even with GPU acceleration. One approach to reduce the number of iterations for CG is to use a preconditioner (*cf.* Section 3.5.2.1). Hence, it would be interesting to devise preconditioners that work well for the multiplicative HQ form.

Another strategy is to solve the system of equations with reduced accuracy, thus needing fewer iterations of CG. This has been undertaken by Gilavert et al. [2015], who added an acceptance step to guarantee that the sampling approach still yields samples from the target distribution.

Finally, one may use the additive HQ form instead, which leads to equation systems that are easier to solve. As far as we are aware, the additive form has not been used for sampling-based HQ inference. One reason may be that the resulting Gibbs sampler will likely exhibit poor mixing due to the specific representation in the additive HQ form. Using a pairwise MRF prior from [Schmidt, 2010] (similar to that of [Schmidt et al., 2010]) with an additive HQ representation, we found in preliminary experiments (not discussed in this thesis) that mixing is indeed much slower. Concretely, while we performed a few hundred iterations of Gibbs sampling with the multiplicative HQ form (as in Chapter 4), we instead required several thousand – but much faster – iterations of the Gibbs sampler to obtain similar results when using the same potential represented in the additive form. Nevertheless, in both cases we were able to reach similar results w.r.t. the quality of the restored images. Overall, we found in our preliminary experiments with a pairwise MRF prior that a representation in the additive form enables faster sampling-based inference as compared to the multiplicative form (*i. e.*, we obtain the same image quality in less time), especially for larger images. Furthermore, it is conceivable that mixing could be improved through *tempering* schemes [*e. g.*, Neal, 1996; Earl and Deem, 2005], such as exchanging the states of several Markov chains that run in parallel at different temperatures.

8.2.3 *Jointly learning model and inference*

It is common to separate a model from the optimization algorithm that is employed for inference. For example, we can use a variety of different algorithms to solve energy minimization problems. As a consequence, models for particular problems, such as image restoration, and optimization / inference algorithms are typically developed independently of each other. For some representative models, the latter are often analyzed in terms of convergence speed to a solution with a particular numerical accuracy.

In Chapters 6 and 7, we took a different approach by combining a model and a particular (iterative) inference algorithm in a single unit. While this has the disadvantage of yielding a highly-specialized method that will not work well for different scenarios, it has two key advantages: First, we can adapt model and inference algorithm to each other, such as removing unnecessary parts or using more advantageous parameterizations. For example, we replaced potentials with shrinkage functions in Chapter 7, which yielded more flexibility and simpler parameter learning. Second, we can learn custom model and optimization parameters for every step of inference, which allows us to achieve good results with only a few iterations. Instead of convergence speed to an energy minimum, we care about achieving a good solution w.r.t. an application-specific loss function (*e. g.*, PSNR) in as few iterations as possible.

More concretely, we proposed a generic discriminative generalization of half-quadratic MAP estimation with a generative model in Chapter 6. In Chapter 7, we took the same general approach, but focused on MAP estimation with a particular additive HQ form, which allows for very fast inference. However, the idea of jointly learning a model and its associated inference algorithm applies much more generally. Therefore, in the future we should investigate other interesting combinations of models and inference methods.

One such model and inference combination was recently proposed by Chen et al. [2015], which can be seen as a combination of our approach from Chapter 7 with that of Barbu [2009]. Concretely, Chen et al. [2015] learn gradient-descent steps similar to [Barbu, 2009], but explain them mostly in the context of reaction-diffusion approaches. In contrast to [Barbu, 2009], they essentially adopt our learning and parameterization approach to learn a tailored model for each gradient step.

8.2.3.1 *Connection with deep neural networks*

Essentially, (supervised) neural networks [*e.g.*, Jain and Seung, 2009; Burger et al., 2012] can also be interpreted as a combination of model and inference. Furthermore, we can relate our approaches from Chapters 6 and 7 to deep neural networks.

In Chapter 6, we proposed a cascade of Gaussian CRFs, where regression trees determined the potentials of the Gaussian CRF at each model stage. Alternatively, we could have used (convolutional) neural networks to regress the parameters of the potentials functions. This would have led to a deep neural network (NN) with the main difference that we carry out Gaussian CRF inference after each layer (stage of the cascade). Nevertheless, such a deep neural network would be differentiable and thus amenable to standard gradient-based training.

Our cascade of shrinkage fields (CSF) model from Chapter 7 is actually a particular deep neural network, where the structure of the deep network is fully determined by the combination of the model and the specific inference algorithm. The interpretation as a NN may be helpful to devise future extensions of our approach. In particular, each stage (or layer) of a CSF first requires the computation of several convolutions, point-wise non-linearities (via shrinkage functions) and additions, as are common in convolutional NNs. However, note that our non-linearities are learned, which is not typical for NNs. Additionally, we need to solve a system of linear equations for Gaussian CRF inference. In case of a CSF, this can be done by element-wise multiplication in Fourier space before transforming the solution back to the spatial domain. As result of the well-known convolution theorem (*cf.* Section 3.5.1.2), this is equivalent to convolution (with a filter as large as the input image). Hence, the inference step of a CSF can equivalently be performed by a NN with dense connectivity. Overall, a cascade of shrinkage fields is a deep neural network, where most layers (feature maps) are sparsely-connected via additions or convolutions with filters of small support, but some layers are fully-connected as they correspond to Gaussian inference as described above. A possible future extension is to replace the fully-connected layers (which correspond to solving the system of equations with via DFTs) by using convolutions with filters of smaller support, which has similarly been done by Badri et al. [2015].

In general, there seems to be a recent trend [*e. g.*, Zheng et al., 2015] to combine and fully integrate two types of approaches: *1)* powerful – but unstructured – prediction functions for regression and classification, such as random forests and especially deep neural networks, and *2)* structured prediction via CRF inference, which allow to model constraints between the output labels.

8.2.4 *Towards richer and more accurate models*

BEYOND GAUSSIAN NOISE Recall that we applied HQ augmentation only to the image prior, which allowed us to obtain a posterior distribution that is Gaussian when conditioned on the added auxiliary variables. This was only possible due to our assumption of a Gaussian likelihood. However, if this were not the case, we may also apply HQ augmentation to the likelihood to obtain a conditionally Gaussian posterior distribution (*cf.* Section 3.2). This immediately allows to address the removal of other types of noise beyond additive Gaussian noise, and could open up HQ inference to other applications.

OTHER PRIOR KNOWLEDGE Currently, MRFs and CRFs for natural images typically model filter responses via heavy-tailed potentials. However, images also contain other statistical regularities. We briefly discussed *self-similarity* in Section 2.6.3, which Sun and Tappen [2011] attempted to integrate with a random field architecture. Nevertheless, there is much more prior knowledge that could be integrated into a model. However, one must strike a balance between expressive models and still being able to carry out inference in a somewhat efficient manner. For example, Mei et al. [2015] recently proposed a method to integrate *marginal histogram constraints* in a way that still allows efficient inference using HQ techniques.

Another approach that we briefly mentioned in Section 3.2 is to connect the latent variables after HQ augmentation. Given that the latent variables approximately indicate edges in the images, this could model hysteresis or non-maximum suppression of edges [*cf.* Black and Rangarajan, 1996]. However, updating the latent variables becomes more complicated when they are no longer independent. Consequently, one has to devise constructions that still allow for efficient optimization.

JOINT MODELS If we consider the visual complexity of the real world, it becomes clear that our image models are still very crude as they do not even try to accurately describe the image formation process. For example, an image could be modeled as a projection of a three-dimensional scene, which is composed of many kinds of objects at different locations that each reflect light from a variety of sources in different ways. Although this is still a simplistic model of a real image, it currently is too complex to allow for efficient inference. Nevertheless, we have to devise better (generative) models of the image formation process in the future. Additionally, discriminative methods can help to improve inference in such models [*e.g.*, Jampani et al., 2015].

There has been some work towards that end, in particular by jointly addressing two (or more) problems that have previously been handled separately. For example, Le Roux et al. [2011] jointly address the appearance and shape of image patches with a masked restricted Boltzmann machine that models occlusion boundaries. Zhang et al. [2011] show that both image restoration and (face) recognition can benefit from each other when performed jointly. Sun et al. [2014] consider the issue that image priors are typically trained with a diverse set of images from a variety of scenes, which means that they can be suboptimal for a specific image; to that end, the adapt the prior based on images of the same scene type.

Although these are promising examples, they are just a start towards richer and more accurate models of visual scenes. There is much work to be done.

APPENDIX A

CONTENTS

A.1 Supplemental Material for Chapter 5 **185**
  A.1.1 Parameter learning in transformation-invariant product models 185
  A.1.2 Details of parameter learning 186
  A.1.3 Details of feature extraction 187
  A.1.4 Additional descriptor details 188
  A.1.5 Experimental details 189
A.2 Supplemental Material for Chapter 7 **190**
  A.2.1 Boundary handling 190
  A.2.2 Learning 192

A.1 SUPPLEMENTAL MATERIAL FOR CHAPTER 5

We provide additional details that are not necessary to understand Chapter 5, but are especially helpful to reproduce the results.

A.1.1 *Parameter learning in transformation-invariant product models*

We briefly demonstrate that parameter learning in product models with integrated transformation invariance (*cf.* Section 5.1.1) involves only minor modifications to existing gradient-based learning algorithms. We begin by considering a generic product model

$$p(\mathbf{x}; \boldsymbol{\Theta}) = \frac{1}{Z(\boldsymbol{\Theta})} \prod_{i=1}^{|\mathcal{F}|} \varphi_i\left(\mathbf{F}_{(i)}\mathbf{x}; \theta_i\right) \tag{A.1}$$

as defined in Eq. (5.1). Now the task is to learn all parameters $\boldsymbol{\Theta} = \{\mathbf{F}_{(i)}, \theta_i | i = 1, \ldots\}$ by using gradients of the log-probability (density) $\log p(\mathbf{x}; \boldsymbol{\Theta})$. It is quite straightforward to see that the partial derivatives of the log-probability (density) w.r.t. the factor parameters θ_i and the feature transformations $\mathbf{F}_{(i)}$ are given as

$$\frac{\partial \log p(\mathbf{x}; \boldsymbol{\Theta})}{\partial \theta_i} = \frac{\partial \log \varphi_i\left(\mathbf{F}_{(i)}\mathbf{x}; \theta_i\right)}{\partial \theta_i} - \frac{\partial \log Z(\boldsymbol{\Theta})}{\partial \theta_i} \tag{A.2}$$

$$\frac{\partial \log p(\mathbf{x}; \boldsymbol{\Theta})}{\partial \mathbf{F}_{(i)}} = \frac{\partial \log \varphi_i\left(\mathbf{F}_{(i)}\mathbf{x}; \theta_i\right)}{\partial \mathbf{F}_{(i)}} - \frac{\partial \log Z(\boldsymbol{\Theta})}{\partial \mathbf{F}_{(i)}}. \tag{A.3}$$

The gradient of the log-partition function $\log Z(\boldsymbol{\Theta})$ (second term of Eqs. (A.2) and (A.3)) is usually approximated by evaluating the first term of the respective equation on a set of samples from the product model (*cf.* Section 2.3.2).

According to Eq. (5.4), we define a transformation-invariant product model w.r.t. a set of linear image transformations $\mathcal{T} = \{\mathbf{T}_{(j)} | j = 1, \dots\}$ as

$$p_{\mathcal{T}}(\mathbf{x}; \boldsymbol{\Theta}) = \frac{1}{Z(\boldsymbol{\Theta})} \prod_{j=1}^{|\mathcal{T}|} \prod_{i=1}^{|\mathcal{F}|} \varphi_i\big(\mathbf{F}_{(i)} \mathbf{T}_{(j)} \mathbf{x}; \theta_i\big). \tag{A.4}$$

In analogy to Eq. (A.1) the partial derivatives of the log-probability (density) thus follow as

$$\begin{aligned} \frac{\partial \log p_{\mathcal{T}}(\mathbf{x}; \boldsymbol{\Theta})}{\partial \theta_i} &= \left[\sum_{j=1}^{|\mathcal{T}|} \frac{\partial \log \varphi_i\big(\mathbf{F}_{(i)} \mathbf{T}_{(j)} \mathbf{x}; \theta_i\big)}{\partial \theta_i} \right] \\ &\quad - \frac{\partial \log Z(\boldsymbol{\Theta})}{\partial \theta_i} \end{aligned} \tag{A.5}$$

$$\begin{aligned} \frac{\partial \log p_{\mathcal{T}}(\mathbf{x}; \boldsymbol{\Theta})}{\partial \mathbf{F}_{(i)}} &= \left[\sum_{j=1}^{|\mathcal{T}|} \frac{\partial \log \varphi_i\big(\mathbf{F}_{(i)} \mathbf{T}_{(j)} \mathbf{x}; \theta_i\big)}{\partial \mathbf{F}_{(i)}} \right] \\ &\quad - \frac{\partial \log Z(\boldsymbol{\Theta})}{\partial \mathbf{F}_{(i)}}. \end{aligned} \tag{A.6}$$

Please note that Eqs. (A.5) and (A.6) involve quite similar derivative calculations as Eqs. (A.2) and (A.3). In essence, the derivatives of the log-factors $\log \varphi_i$ need to be summed over $|\mathcal{T}|$ transformed inputs. Hence, integrating transformation-invariance into existing product model implementations generally requires little implementation effort.

A.1.2 *Details of parameter learning*

A.1.2.1 *R-FoE*

The R-FoE model of Section 5.2 was trained on a database of 5000 natural images (of size 50×50 pixels) using persistent contrastive divergence [Tieleman, 2008]. Learning was done with stochastic gradient descent using mini-batches of 100 images (and model samples) for a total of 10000 (exponentially smoothed) gradient steps with an annealed learning rate. We trained the model using conditional sampling to avoid boundary issues [Norouzi et al., 2009]. Both learned filters were initialized randomly from a standard normal distribution, and constrained to have mean 0 and norm 1 throughout learning. We initialized the shapes of the potential functions to be very broad (*cf.* [Gao and Roth, 2012]).

A.1.2.2 *RC-RBM*

We trained our RC-RBM from Section 5.3 akin to the algorithm of [Norouzi et al., 2009], in particular using contrastive divergence [Hinton, 2002] with one step of Gibbs sampling, although applying Rao-Blackwellization [*cf.* Swersky et al., 2010] to minimize sample variance. We used two datasets for unsupervised training (always using one visible unit per image pixel): a random subset of 10000 binary images from the training set of the MNIST handwritten digits [LeCun and Cortes], and the same dataset of natural images as used for the R-FoE, but here ZCA-whitened [*cf.* Hyvärinen et al., 2009, § 5]. In both cases, we performed stochastic gradient descent with mini-batches of 20 images (100 for MNIST), an annealed learning rate, and exponential gradient smoothing. For training on natural images, we also relied on conditional sampling to avoid boundary issues [Norouzi et al., 2009].

We initialized all hidden biases to $\mathbf{b} = -\mathbf{3}$ and all visible biases to $\mathbf{c} = \mathbf{0}$; note that training on MNIST relied on individual biases $\mathbf{c}$, and training on natural images used a shared (scalar) bias c for all visible units. Instead of fixing the hidden biases $\mathbf{b}$ to a small value to encourage sparsity [Norouzi et al., 2009], we learned them together with the features, which we constrained to have the same norm, updated slowly over time through exponential smoothing. We did not use any additional regularization terms to encourage learning of sparse features (such as [Lee et al., 2008]).

We only define the filters $\mathbf{w}_i$ inside a circular area by actually using $\hat{\mathbf{R}}_{(\omega)} = \mathbf{B} \cdot \mathbf{R}_{(\omega)}$ instead of $\mathbf{R}_{(\omega)}$ in Eq. (5.12), where multiplication with $\mathbf{B}$ extracts the circular interior of the image patch as a vector.

A.1.3 *Details of feature extraction*

We extract RC-RBM features by computing the hidden activation probabilities $p_{\text{RC-RBM}}(\mathbf{h} = \mathbf{1}|\mathbf{x})$ for each feature i convolutionally at all image locations (k,l) and all specified rotation angles ω. Computation is straightforward since $p_{\text{RC-RBM}}(\mathbf{h} = \mathbf{1}|\mathbf{x})$ decomposes into a product of univariate distributions

$$p_{\text{RC-RBM}}(h_{(\omega),(k,l),i} = 1|\mathbf{x}) = \operatorname{sig}(\mathbf{w}_i^{\mathsf{T}} \mathbf{R}_{(\omega)} \mathbf{C} \mathbf{S}_{(k,l)} \mathbf{x} + b_i) \tag{A.7}$$

with the logistic function $\operatorname{sig}(x) = 1/(1 + e^{-x})$. We note that we set all hidden biases $\mathbf{b} = \mathbf{0}$ for feature extraction after learning, as this significantly increased performance in recognition and detection tasks. Furthermore, non-maximum suppression (over rotations ω) is used to only retain activations with maximum probability at each location (k,l). The EHOF/IHOF descriptor is computed separately for each (of the four) learned features; the final descriptor is then obtained by concatenation of the individual descriptor vectors.

Oriented gradient features are extracted the same way as in the popular HOG descriptor [Dalal and Triggs, 2005]: Centered image derivatives ($[1, 0, -1]$ and $[1, 0, -1]^T$) are first computed at all image locations to obtain horizontal and vertical derivative images. Each pixel is then assigned to one of B orientation angles (using linear interpolation) according to its gradient angle and represented by its gradient magnitude.

A.1.4 *Additional descriptor details*

A.1.4.1 *Local cell normalization*

When using oriented gradient features, we also perform two different normalizations of all cells (except the central one), akin to the block normalization procedure in HOG. We do this because it significantly increases performance.

Here, each block consists of two neighboring cells on a ring, *i.e.* each cell is normalized with its predecessor and successor cell. We use L2-normalization $g(\mathbf{v}, \mathbf{z}) = \mathbf{v} / \sqrt{\|\mathbf{z}\|^2 + \epsilon}$ for cell histogram vector $\mathbf{v}$ with block vector $\mathbf{z}$ and $\epsilon = 10^{-4}$; we do not "clip" values after normalization. The layout in the descriptor matrix is adjusted in order to retain the equivariance property: Different normalizations of the same cell are (deterministically) grouped together in the columns of the matrix, *i.e.* first come all (both) normalizations of the normalized entry for orientation angle 1, then all normalizations for angle 2, *etc.*

Unfortunately, this local cell normalization procedure does not improve results when applied to our learned RC-RBM features, hence we do not use it; however, we can assume that a suitable normalization scheme would also enhance the performance for our learned features, but we have not explored this yet. In this sense, gradient features are at an advantage due to previous research on suitable normalizations, which we leverage here.

A.1.4.2 *Descriptor dimensionality and scaling*

For an EHOF descriptor with R rings, C cells per ring, and features extracted at O orientations, we obtain a 3-dimensional histogram $\mathbf{H}_3 \in \mathbb{R}^{R \times C \times O}$, which is reshaped into the 2-dimensional $\mathbf{H}_2 \in \mathbb{R}^{R \cdot C \times n \cdot O}$. For gradient features, $n = 2$ due to the two normalizations of each cell, and $n = 1$ for our learned RC-RBM features. Hence, we obtain the descriptor dimensionality $D = R \cdot C \cdot n \cdot O + O$, where the last term comes from the histogram vector of central cell $\mathbf{c} \in \mathbb{R}^O$. For IHOF, we additionally compute the 2D-DFT of $\mathbf{H}_2$ and the 1D-DFT of $\mathbf{c}$ and only retain the magnitude in both cases. The DFT magnitude of real inputs exhibits redundancies, which we have only removed in case of 1D-DFT, resulting in the IHOF descriptor dimensionality $D = R{\cdot}C{\cdot}n{\cdot}O + \lceil \frac{O}{2} \rceil$.

The resulting descriptor vectors are scaled to unit infinity norm in case of IHOF (and EHOF for car detection).

A.1.5 *Experimental details*

A.1.5.1 *Denoising*

For the denoising results in Section 5.5, we used a fixed sampling scheme similar to [Schmidt et al., 2010] with four independent samplers, each running for 60 iterations to yield 120 samples overall after discarding 30 burn-in iterations each. We employed the same procedure for the denoising example in Fig. 5.5, but used 240 samples in total to further reduce to variability induced by the sampling process.

A.1.5.2 *Car detection*

RC-RBM features were extracted from whitened grayscale versions of the RGB color input images, whereas gradient features were obtained by taking the maximum gradient response (in terms of magnitude) over the three channels of the color images. For the HOG baseline performance, we use the implementation of [Dollár] with 5×5 pixel-sized cells and 9 (unsigned) orientation angles.

We trained the linear SVM initially by performing cross-validation to find the best regularization parameter C, then used two rounds of bootstrapping to obtain the final model. Detection was carried out with a search stride of 5 pixels over five image scales

$$[1.0^{-1}, 1.1^{-1}, 1.2^{-1}, 1.3^{-1}, 1.4^{-1}], \tag{A.8}$$

where 1.0 refers to the original image size. We evaluated the performance according to the PASCAL VOC 2010 criteria [Everingham et al., 2010], *i. e.* requiring 50% overlap with a ground-truth annotation and not allowing multiple detections of the same car. Calculation of the average precision also followed [Everingham et al., 2010].

The following derivations require only standard (multivariate) calculus, where we use the *numerator layout notation*[1]. Please note that all derived formulae can be computed efficiently and implemented compactly (MATLAB code for learning and inference is available online).

We derive the iterative update equation for the image variables $\mathbf{x}$ in Eq. (7.6) as:

$$
\begin{aligned}
g_\beta(\mathbf{z}) &= \arg\min_{\mathbf{x}} E_\beta(\mathbf{x}|\mathbf{z},\mathbf{y}) \\
&= \arg\min_{\mathbf{x}} \frac{\lambda}{2}\|\mathbf{y}-\mathbf{K}\mathbf{x}\|^2 + \sum_{i=1}^{N}\sum_{c\in\mathcal{C}}\left(\frac{\beta}{2}\left(\mathbf{f}_i^\mathsf{T}\mathbf{x}_{(c)} - z_{ic}\right)^2 + \rho_i(z_{ic})\right) \\
&= \arg\min_{\mathbf{x}} \frac{\lambda}{2}\|\mathbf{y}-\mathbf{K}\mathbf{x}\|^2 + \frac{\beta}{2}\sum_{i=1}^{N}\|\mathbf{F}_i\mathbf{x}-\mathbf{z}_i\|^2 \\
&= \arg\min_{\mathbf{x}} \frac{1}{2}\mathbf{x}^\mathsf{T}\left[\lambda\mathbf{K}^\mathsf{T}\mathbf{K} + \beta\sum_{i=1}^{N}\mathbf{F}_i^\mathsf{T}\mathbf{F}_i\right]\mathbf{x} - \mathbf{x}^\mathsf{T}\left[\lambda\mathbf{K}^\mathsf{T}\mathbf{y} + \beta\sum_{i=1}^{N}\mathbf{F}_i^\mathsf{T}\mathbf{z}_i\right] \\
&= \left[\lambda\mathbf{K}^\mathsf{T}\mathbf{K} + \beta\sum_{i=1}^{N}\mathbf{F}_i^\mathsf{T}\mathbf{F}_i\right]^{-1}\left[\lambda\mathbf{K}^\mathsf{T}\mathbf{y} + \beta\sum_{i=1}^{N}\mathbf{F}_i^\mathsf{T}\mathbf{z}_i\right] \\
&= \left[\frac{\lambda}{\beta}\mathbf{K}^\mathsf{T}\mathbf{K} + \sum_{i=1}^{N}\mathbf{F}_i^\mathsf{T}\mathbf{F}_i\right]^{-1}\left[\frac{\lambda}{\beta}\mathbf{K}^\mathsf{T}\mathbf{y} + \sum_{i=1}^{N}\mathbf{F}_i^\mathsf{T}\mathbf{z}_i\right] \\
&= \mathcal{F}^{-1}\left[\frac{\mathcal{F}\left(\frac{\lambda}{\beta}\mathbf{K}^\mathsf{T}\mathbf{y} + \sum_{i=1}^{N}\mathbf{F}_i^\mathsf{T}\mathbf{z}_i\right)}{\frac{\lambda}{\beta}|\check{\mathbf{K}}|^2 + \sum_{i=1}^{N}|\check{\mathbf{F}}_i|^2}\right]. \qquad \text{(A.9)}
\end{aligned}
$$

Convolution is again denoted by $\mathbf{F}\mathbf{x} = [\mathbf{f}^\mathsf{T}\mathbf{x}_{(\mathcal{C}_1)}, \ldots, \mathbf{f}^\mathsf{T}\mathbf{x}_{(\mathcal{C}_{|\mathcal{C}|})}]^\mathsf{T} \equiv \mathbf{f} \otimes \mathbf{x} \equiv \mathcal{F}^{-1}(\check{\mathbf{F}} \circ \mathcal{F}(\mathbf{x}))$. The optical transfer function $\check{\mathbf{F}} \equiv \mathcal{F}(\mathbf{f})$ is derived from filter (point spread function) $\mathbf{f}$, where $\mathcal{F}$ denotes the discrete Fourier transform (DFT). Note that division is applied element-wise in Eq. (A.9). The last step is possible since $\mathbf{K}$ and the matrices $\mathbf{F}_i$ are BCCB as they correspond to convolutions with circular boundary conditions. Consequently, they are diagonalized by DFTs, allowing element-wise operations.

A.2.1 *Boundary handling*

To reduce boundary artifacts, which may arise due to using convolutions with circular boundary conditions, we use padding of the input image. Specifically, we first take the input image $\mathbf{x} \in \mathbb{R}^D$ (of width w and height h with $D = h \cdot w$) and replicate b pixels of its boundary on all sides by multiplication with the sparse "padding" matrix $\mathbf{P}_b$[2]. In

---

[1] *cf.* http://en.wikipedia.org/w/index.php?title=Matrix_calculus&oldid=576691987#Numerator-layout_notation.

[2] In MATLAB, $\mathbf{P}_b\mathbf{x} \equiv$ `reshape(padarray(reshape(x,h,w),[b,b],'replicate','both'),D,1)`.

image denoising, b can be freely chosen (we use $b = 10$); for deconvolution $b = (r-1)/2$ with square blur kernel $\mathbf{k} \in \mathbb{R}^{r^2}$ of size $r \times r$ pixels[3].

For deconvolution we additionally apply "edge-tapering" to the padded input image $\mathbf{P}_b\mathbf{x}$ so that it better matches the assumptions of applying convolution with circular boundary handling. In particular, we applied u iterations of MATLAB's `edgetaper` function (where $\boldsymbol{\alpha}_\mathbf{k}$ is a weighting vector based on $\mathbf{k}$), which can be formalized as

$$\begin{aligned}
\texttt{edgetaper}(\mathbf{x}, \mathbf{k}) &= \boldsymbol{\alpha}_\mathbf{k} \circ \mathbf{x} + (\mathbf{1} - \boldsymbol{\alpha}_\mathbf{k}) \circ (\mathbf{k} \otimes \mathbf{x}) && \text{(A.10)} \\
&= \mathcal{D}\{\boldsymbol{\alpha}_\mathbf{k}\}\mathbf{x} + \mathcal{D}\{\mathbf{1} - \boldsymbol{\alpha}_\mathbf{k}\}(\mathbf{K}\mathbf{x}) && \text{(A.11)} \\
&= \big(\mathcal{D}\{\boldsymbol{\alpha}_\mathbf{k}\} + \mathcal{D}\{\mathbf{1} - \boldsymbol{\alpha}_\mathbf{k}\}\mathbf{K}\big)\mathbf{x} && \text{(A.12)} \\
&= \mathbf{E}_\mathbf{k}\mathbf{x}. && \text{(A.13)}
\end{aligned}$$

Performing u iterations of edge-tapering can thus be expressed by multiplication with the matrix $\mathbf{E}_\mathbf{k} = \mathcal{D}\{\boldsymbol{\alpha}_\mathbf{k}\} + \mathcal{D}\{\mathbf{1} - \boldsymbol{\alpha}_\mathbf{k}\}\mathbf{K}$ raised to the u^{th} power. (For denoising, $\mathbf{E}_\mathbf{k}^u = \mathbf{I}$ is set to the identity matrix.)

We denote the boundary-padded and possibly edge-tapered input image as $\mathbf{x}^\text{P} = \mathbf{E}_\mathbf{k}^u \mathbf{P}_b \mathbf{x}$, and as $\hat{\mathbf{x}}^\text{B}$ the output image where the padded boundary region has not been removed yet. Consequently, we use the following minor variation of $g_{\boldsymbol{\Theta}}(\mathbf{x})$ (Eqs. 7.17 and 7.18) in practice:

$$\begin{aligned}
g_{\boldsymbol{\Theta}}(\mathbf{x}) &= \mathbf{T}_b\, \mathcal{F}^{-1} \left[\frac{\mathcal{F}\left(\lambda \mathbf{K}^\mathsf{T}\mathbf{y} + \sum_{i=1}^N \mathbf{F}_i^\mathsf{T} f_{\pi_i}(\mathbf{F}_i \mathbf{x}^\text{P})\right)}{\lambda |\check{\mathbf{K}}|^2 + \sum_{i=1}^N |\check{\mathbf{F}}_i|^2} \right] && \text{(A.14)} \\
&= \mathbf{T}_b \underbrace{\left[\lambda \mathbf{K}^\mathsf{T}\mathbf{K} + \sum_{i=1}^N \mathbf{F}_i^\mathsf{T}\mathbf{F}_i\right]^{-1}}_{\boldsymbol{\Omega}^{-1}} \underbrace{\left[\lambda \mathbf{K}^\mathsf{T}\mathbf{y} + \sum_{i=1}^N \mathbf{F}_i^\mathsf{T} f_{\pi_i}(\mathbf{F}_i \mathbf{x}^\text{P})\right]}_{\boldsymbol{\eta}} && \text{(A.15)} \\
&= \mathbf{T}_b \hat{\mathbf{x}}^\text{B}. && \text{(A.16)}
\end{aligned}$$

The uncropped output image is given as $\hat{\mathbf{x}}^\text{B} = \boldsymbol{\Omega}^{-1}\boldsymbol{\eta}$. To obtain an output image $\hat{\mathbf{x}} = g_{\boldsymbol{\Theta}}(\mathbf{x})$ with the same size as the input image $\mathbf{x}$, we remove the padded boundary region by multiplication with the sparse ("cropping") matrix $\mathbf{T}_b$.

It is important to note that padding, edge-tapering, and cropping can all be expressed as linear transformations, which allows us to easily compute gradients of the transformed images. Furthermore, please note that for inference and learning (Section A.2.2), the convolution matrices $\mathbf{K}$ and $\mathbf{F}_i$ are never explicitly constructed (also applies to $\mathbf{K}^\mathsf{T}$ and $\mathbf{F}_i^\mathsf{T}$), since all matrix-vector products can be efficiently computed through convolutions.

---

[3] Square kernel only assumed for simplicity here.

A.2.2 *Learning*

In the following we use $g_{\boldsymbol{\Theta}}(\mathbf{x})$ with boundary handling as defined in Eq. (A.16). Recall from Chapter 7 that given training data $\{\mathbf{x}_{\mathrm{gt}}^{(s)}, \mathbf{y}^{(s)}, \mathbf{k}^{(s)}\}_{s=1}^{S}$, we learn all model parameters $\boldsymbol{\Theta}_t = \{\lambda_t, \boldsymbol{\pi}_{ti}, \mathbf{f}_{ti}\}_{i=1}^{N}$ greedily stage-by-stage for $t = 1, \ldots, T$ via

$$J(\boldsymbol{\Theta}_t) = \sum_{s=1}^{S} \ell(\hat{\mathbf{x}}_t^{(s)}; \mathbf{x}_{\mathrm{gt}}^{(s)}), \tag{A.17}$$

or jointly for all T stages via

$$J(\boldsymbol{\Theta}_{1,\ldots,T}) = \sum_{s=1}^{S} \ell(\hat{\mathbf{x}}_T^{(s)}; \mathbf{x}_{\mathrm{gt}}^{(s)}). \tag{A.18}$$

To that end, at each stage of greedy learning $J(\boldsymbol{\Theta}_t)$ we need to compute Eq. (A.17) and its gradient $\frac{\partial J(\boldsymbol{\Theta}_t)}{\partial \boldsymbol{\Theta}_t}$; for joint training $J(\boldsymbol{\Theta}_{1,\ldots,T})$ we need to compute Eq. (A.18) and its gradient $\left[\frac{\partial J(\boldsymbol{\Theta}_{1,\ldots,T})}{\partial \boldsymbol{\Theta}_1}, \ldots, \frac{\partial J(\boldsymbol{\Theta}_{1,\ldots,T})}{\partial \boldsymbol{\Theta}_T}\right]$.

A.2.2.1 *Gradient of loss function*

We can address one training example at a time since the gradients of Eqs. (A.17) and (A.18) decompose into sums over training examples. The gradient w.r.t. the model parameters of the last/current stage can be computed as follows:

$$\begin{aligned}
\frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\mathrm{gt}})}{\partial \boldsymbol{\Theta}_t} &= \frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\mathrm{gt}})}{\partial \hat{\mathbf{x}}_t} \cdot \frac{\partial \mathbf{T}_b \boldsymbol{\Omega}_t^{-1} \boldsymbol{\eta}_t}{\partial \boldsymbol{\Theta}_t} && \text{(A.19)}\\
&= \frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\mathrm{gt}})}{\partial \hat{\mathbf{x}}_t} \mathbf{T}_b \boldsymbol{\Omega}_t^{-1} \left[-\frac{\partial \boldsymbol{\Omega}_t}{\partial \boldsymbol{\Theta}_t} \boldsymbol{\Omega}_t^{-1} \boldsymbol{\eta}_t + \frac{\partial \boldsymbol{\eta}_t}{\partial \boldsymbol{\Theta}_t} \right] && \text{(A.20)}\\
&= \hat{\mathbf{c}}_t^{\mathsf{T}} \left[\frac{\partial \boldsymbol{\eta}_t}{\partial \boldsymbol{\Theta}_t} - \frac{\partial \boldsymbol{\Omega}_t}{\partial \boldsymbol{\Theta}_t} \hat{\mathbf{x}}_t^{\mathrm{B}} \right] \quad \text{with} \quad \hat{\mathbf{c}}_t = \boldsymbol{\Omega}_t^{-1} \mathbf{T}_b^{\mathsf{T}} \left[\frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\mathrm{gt}})}{\partial \hat{\mathbf{x}}_t} \right]^{\mathsf{T}}. && \text{(A.21)}
\end{aligned}$$

The gradient of the chosen loss function (negative PSNR)

$$\ell(\hat{\mathbf{x}}; \mathbf{x}_{\mathrm{gt}}) = -20 \log_{10} \left(\frac{R\sqrt{D}}{\|\hat{\mathbf{x}} - \mathbf{x}_{\mathrm{gt}}\|} \right), \tag{A.22}$$

where D denotes the number of pixels of $\hat{\mathbf{x}}$ and R the maximum intensity level of a pixel (*i.e.*, $R = 255$), is derived as

$$\begin{aligned}
\frac{\partial \ell(\hat{\mathbf{x}}; \mathbf{x}_{\mathrm{gt}})}{\partial \hat{\mathbf{x}}} &= -\frac{20}{\log 10} \frac{\partial}{\partial \hat{\mathbf{x}}} \log \left(\frac{R\sqrt{D}}{\|\hat{\mathbf{x}} - \mathbf{x}_{\mathrm{gt}}\|} \right) && \text{(A.23)}\\
&= \frac{20}{\log 10} \frac{\partial \log \|\hat{\mathbf{x}} - \mathbf{x}_{\mathrm{gt}}\|}{\partial \hat{\mathbf{x}}} && \text{(A.24)}\\
&= \frac{20}{\log 10} \frac{1}{\|\hat{\mathbf{x}} - \mathbf{x}_{\mathrm{gt}}\|} \frac{\partial \|\hat{\mathbf{x}} - \mathbf{x}_{\mathrm{gt}}\|}{\partial \hat{\mathbf{x}}} && \text{(A.25)}\\
&= \frac{20}{\log 10} \frac{(\hat{\mathbf{x}} - \mathbf{x}_{\mathrm{gt}})^{\mathsf{T}}}{\|\hat{\mathbf{x}} - \mathbf{x}_{\mathrm{gt}}\|^2}. && \text{(A.26)}
\end{aligned}$$

For joint training we additionally require gradients w.r.t. the model parameters of previous stages. The gradient w.r.t. the second-to-last stage is obtained as ($t \geq 2$)

$$\begin{aligned}
\frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\text{gt}})}{\partial \boldsymbol{\Theta}_{t-1}} &= \hat{\mathbf{c}}_t^{\mathsf{T}} \left[\frac{\partial \boldsymbol{\eta}_t}{\partial \boldsymbol{\Theta}_{t-1}} \right] \\
&= \hat{\mathbf{c}}_t^{\mathsf{T}} \frac{\partial}{\partial \boldsymbol{\Theta}_{t-1}} \left(\sum_{i=1}^{N} \mathbf{F}_{ti}^{\mathsf{T}} f_{\boldsymbol{\pi}_{ti}}(\mathbf{F}_{ti} \hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) \right) \\
&= \hat{\mathbf{c}}_t^{\mathsf{T}} \sum_{i=1}^{N} \mathbf{F}_{ti}^{\mathsf{T}} f_{\boldsymbol{\pi}_{ti}}'(\mathbf{F}_{ti} \hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) \mathbf{F}_{ti} \mathbf{E}_{\mathbf{k}}^{u} \mathbf{P}_b \frac{\partial \hat{\mathbf{x}}_{t-1}}{\partial \boldsymbol{\Theta}_{t-1}} \\
&= \left[\sum_{i=1}^{N} (\mathbf{F}_{ti} \hat{\mathbf{c}}_t)^{\mathsf{T}} f_{\boldsymbol{\pi}_{ti}}'(\mathbf{F}_{ti} \hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) \mathbf{F}_{ti} \right] \mathbf{E}_{\mathbf{k}}^{u} \mathbf{P}_b \frac{\partial \mathbf{T}_b \boldsymbol{\Omega}_{t-1}^{-1} \boldsymbol{\eta}_{t-1}}{\partial \boldsymbol{\Theta}_{t-1}} \\
&= \left[\mathbf{T}_b^{\mathsf{T}} \mathbf{P}_b^{\mathsf{T}} \mathbf{E}_{\mathbf{k}}^{u\,\mathsf{T}} \sum_{i=1}^{N} \mathbf{F}_{ti}^{\mathsf{T}} f_{\boldsymbol{\pi}_{ti}}'(\mathbf{F}_{ti} \hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) \mathbf{F}_{ti} \hat{\mathbf{c}}_t \right]^{\mathsf{T}} \boldsymbol{\Omega}_{t-1}^{-1} \left[\frac{\partial \boldsymbol{\eta}_{t-1}}{\partial \boldsymbol{\Theta}_{t-1}} - \frac{\partial \boldsymbol{\Omega}_{t-1}}{\partial \boldsymbol{\Theta}_{t-1}} \hat{\mathbf{x}}_{t-1}^{\mathrm{B}} \right] \\
&= \hat{\mathbf{c}}_{t-1}^{\mathsf{T}} \left[\frac{\partial \boldsymbol{\eta}_{t-1}}{\partial \boldsymbol{\Theta}_{t-1}} - \frac{\partial \boldsymbol{\Omega}_{t-1}}{\partial \boldsymbol{\Theta}_{t-1}} \hat{\mathbf{x}}_{t-1}^{\mathrm{B}} \right] \qquad \text{(A.27)} \\
&\text{with } \hat{\mathbf{c}}_{t-1} = \boldsymbol{\Omega}_{t-1}^{-1} \mathbf{T}_b^{\mathsf{T}} \mathbf{P}_b^{\mathsf{T}} \mathbf{E}_{\mathbf{k}}^{u\,\mathsf{T}} \sum_{i=1}^{N} \mathbf{F}_{ti}^{\mathsf{T}} f_{\boldsymbol{\pi}_{ti}}'(\mathbf{F}_{ti} \hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) \mathbf{F}_{ti} \hat{\mathbf{c}}_t,
\end{aligned}$$

where $f_{\boldsymbol{\pi}_{ti}}'(\mathbf{F}_{ti} \hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) = \frac{\partial f_{\boldsymbol{\pi}_{ti}}(\mathbf{F}_{ti} \hat{\mathbf{x}}_{t-1}^{\mathrm{P}})}{\partial \mathbf{F}_{ti} \hat{\mathbf{x}}_{t-1}^{\mathrm{P}}}$ (*cf.* Section A.2.2.2). Note that the form is the same as of Eq. (A.21), which means we can apply this recursively to compute gradients for earlier stages ($t \geq 2, s = 1, \ldots, t-1$):

$$\begin{aligned}
\frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\text{gt}})}{\partial \boldsymbol{\Theta}_{t-s}} &= \hat{\mathbf{c}}_{t-s}^{\mathsf{T}} \left[\frac{\partial \boldsymbol{\eta}_{t-s}}{\partial \boldsymbol{\Theta}_{t-s}} - \frac{\partial \boldsymbol{\Omega}_{t-s}}{\partial \boldsymbol{\Theta}_{t-s}} \hat{\mathbf{x}}_{t-s}^{\mathrm{B}} \right] \text{ with} \qquad \text{(A.28)} \\
\hat{\mathbf{c}}_{t-s} &= \boldsymbol{\Omega}_{t-s}^{-1} \mathbf{T}_b^{\mathsf{T}} \mathbf{P}_b^{\mathsf{T}} \mathbf{E}_{\mathbf{k}}^{u\,\mathsf{T}} \sum_{i=1}^{N} \mathbf{F}_{t-s+1,i}^{\mathsf{T}} f_{\boldsymbol{\pi}_{t-s+1,i}}'(\mathbf{F}_{t-s+1,i} \hat{\mathbf{x}}_{t-s}^{\mathrm{P}}) \mathbf{F}_{t-s+1,i} \hat{\mathbf{c}}_{t-s+1}.
\end{aligned}$$

Note that jointly training T stages only takes approximately T times as long as training a single stage.

A.2.2.2 *Gradients for specific model parameters*

Now that we have derived the generic gradients for the given loss function, we need to derive the specific gradients w.r.t. the actual model parameters $\boldsymbol{\Theta}_t = \{\lambda_t, \boldsymbol{\pi}_{ti}, \mathbf{f}_{ti}\}_{i=1}^{N}$ at all stages t.

REGULARIZATION WEIGHT λ We define $\lambda_t = \exp(\tilde{\lambda}_t)$ to ensure positive values of λ_t and learn $\tilde{\lambda}_t$ via its partial derivative

$$\begin{aligned}
\frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\text{gt}})}{\partial \tilde{\lambda}_t} &= \frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\text{gt}})}{\partial \lambda_t} \cdot \frac{\partial \exp(\tilde{\lambda}_t)}{\partial \tilde{\lambda}_t} \qquad \text{(A.29)} \\
&= \hat{\mathbf{c}}_t^{\mathsf{T}} \left(\mathbf{K}^{\mathsf{T}} \mathbf{y} - \mathbf{K}^{\mathsf{T}} \mathbf{K} \hat{\mathbf{x}}_t^{\mathrm{B}} \right) \cdot \lambda_t \qquad \text{(A.30)} \\
&= \lambda_t (\mathbf{K} \hat{\mathbf{c}}_t)^{\mathsf{T}} (\mathbf{y} - \mathbf{K} \hat{\mathbf{x}}_t^{\mathrm{B}}). \qquad \text{(A.31)}
\end{aligned}$$

SHRINKAGE FUNCTION Recall that the shrinkage function is modeled as a linear combination of Gaussian RBF kernels:

$$f_\pi(v) = \sum_{j=1}^{M} \pi_j \exp\left(-\frac{\gamma}{2}(v-\mu_j)^2\right). \tag{A.32}$$

Its derivatives w.r.t. the input and weights can be computed as

$$\frac{\partial f_\pi(v)}{\partial \pi_j} = \exp\left(-\frac{\gamma}{2}(v-\mu_j)^2\right) \tag{A.33}$$

$$\frac{\partial f_\pi(v)}{\partial v} = -\sum_{j=1}^{M} \pi_j \exp\left(-\frac{\gamma}{2}(v-\mu_j)^2\right) \cdot \gamma(v-\mu_j). \tag{A.34}$$

Concerning vector-valued inputs $\mathbf{v} = [v_1, \ldots, v_L]^\mathsf{T}$ to the univariate shrinkage function, please note that

$$f_\pi(\mathbf{v}) = [f_\pi(v_1), \ldots, f_\pi(v_L)]^\mathsf{T} \tag{A.35}$$

$$\frac{\partial f_\pi(\mathbf{v})}{\partial \mathbf{v}} = \mathcal{D}\left\{\frac{\partial f_\pi(v_1)}{\partial v_1}, \ldots, \frac{\partial f_\pi(v_L)}{\partial v_L}\right\} \overset{\text{def}}{=} f'_\pi(\mathbf{v}) \in \mathbb{R}^{L\times L}. \tag{A.36}$$

For practical purposes, we not only precompute and store in a lookup-table (LUT) the shrinkage function $f_\pi(v)$ for all (sensible) v, but all its derivatives that are required for learning the model, which are quickly retrieved from the LUT via linear interpolation.

Concretely, the relevant gradient of the shrinkage function weights is computed as

$$\frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\text{gt}})}{\partial \boldsymbol{\pi}_{ti}} = \hat{\mathbf{c}}_t^\mathsf{T}\left(\mathbf{F}_{ti}^\mathsf{T} \frac{\partial f_{\boldsymbol{\pi}_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}})}{\partial \boldsymbol{\pi}_{ti}}\right) \tag{A.37}$$

$$= (\mathbf{F}_{ti}\hat{\mathbf{c}}_t)^\mathsf{T} \frac{\partial f_{\boldsymbol{\pi}_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}})}{\partial \boldsymbol{\pi}_{ti}}, \tag{A.38}$$

where $\frac{\partial f_{\boldsymbol{\pi}_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}})}{\partial \boldsymbol{\pi}_{ti}} \in \mathbb{R}^{D\times M}$ is a matrix, which contains in each row the derivatives w.r.t. all M entries of vector $\boldsymbol{\pi}_{ti}$ for all D filter responses $\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}$.

FILTER We define each filter of size $m\times n$ w.r.t. a basis $\mathbf{B} \in \mathbb{R}^{mn\times V}$ as $\mathbf{f} = \mathbf{B}\tilde{\mathbf{f}}$ and learn the entries of $\tilde{\mathbf{f}} \in \mathbb{R}^V$. We follow Chen et al. [2013] and choose DCT filters for $\mathbf{B}$, omitting the DC-component to guarantee zero-mean filters.

First, it is useful to denote $\mathbf{f} \otimes \mathbf{x} \equiv \mathbf{F}\mathbf{x} = (\mathbf{f}^\mathsf{T}[\mathbf{x}]_\mathcal{C})^\mathsf{T} = [\mathbf{x}]_\mathcal{C}^\mathsf{T}\mathbf{f}$, where $[\mathbf{x}]_\mathcal{C} \in \mathbb{R}^{mn\times D}$ is a matrix of all cliques $\mathcal{C}$ of $\mathbf{x} \in \mathbb{R}^D$ that filter $\mathbf{f}$ is applied to. Then, we can differentiate the convolved image w.r.t. the filter coefficients $\tilde{\mathbf{f}}$ as follows

$$\frac{\partial \mathbf{F}\mathbf{x}}{\partial \tilde{\mathbf{f}}} = \frac{\partial [\mathbf{x}]_\mathcal{C}^\mathsf{T}\mathbf{B}\tilde{\mathbf{f}}}{\partial \tilde{\mathbf{f}}} = [\mathbf{x}]_\mathcal{C}^\mathsf{T}\mathbf{B} \in \mathbb{R}^{D\times V} \tag{A.39}$$

The gradient of the loss w.r.t. the filter coefficients is derived as

$$
\begin{aligned}
\frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\text{gt}})}{\partial \tilde{\mathbf{f}}_{ti}} &= \hat{\mathbf{c}}_t^\top \left[\frac{\partial \mathbf{F}_{ti}^\top f_{\pi_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}})}{\partial \tilde{\mathbf{f}}_{ti}} - \frac{\partial \mathbf{F}_{ti}^\top \mathbf{F}_{ti}}{\partial \tilde{\mathbf{f}}_{ti}} \hat{\mathbf{x}}_t^{\mathrm{B}} \right] \\
&= f_{\pi_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}})^\top \frac{\partial \mathbf{F}_{ti}\hat{\mathbf{c}}_t}{\partial \tilde{\mathbf{f}}_{ti}} \\
&\quad + (\mathbf{F}_{ti}\hat{\mathbf{c}}_t)^\top \left(\frac{\partial f_{\pi_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}})}{\partial \mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}} \cdot \frac{\partial \mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}}{\partial \tilde{\mathbf{f}}_{ti}} \right) - \frac{\partial (\mathbf{F}_{ti}\hat{\mathbf{c}}_t)^\top \mathbf{F}_{ti}\hat{\mathbf{x}}_t^{\mathrm{B}}}{\partial \tilde{\mathbf{f}}_{ti}} \\
&= f_{\pi_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}})^\top [\hat{\mathbf{c}}_t]_{\mathcal{C}}^\top \mathbf{B} + (\mathbf{F}_{ti}\hat{\mathbf{c}}_t)^\top f_{\pi_{ti}}'(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) [\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}]_{\mathcal{C}}^\top \mathbf{B} \\
&\qquad - (\mathbf{F}_{ti}\hat{\mathbf{c}}_t)^\top [\hat{\mathbf{x}}_t^{\mathrm{B}}]_{\mathcal{C}}^\top \mathbf{B} - (\mathbf{F}_{ti}\hat{\mathbf{x}}_t^{\mathrm{B}})^\top [\hat{\mathbf{c}}_t]_{\mathcal{C}}^\top \mathbf{B} \\
&= \Big[[\hat{\mathbf{c}}_t]_{\mathcal{C}} \left(f_{\pi_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) - \mathbf{F}_{ti}\hat{\mathbf{x}}_t^{\mathrm{B}} \right) \\
&\qquad + [\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}]_{\mathcal{C}} f_{\pi_{ti}}'(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) \mathbf{F}_{ti}\hat{\mathbf{c}}_t - [\hat{\mathbf{x}}_t^{\mathrm{B}}]_{\mathcal{C}} \mathbf{F}_{ti}\hat{\mathbf{c}}_t \Big]^\top \mathbf{B}.
\end{aligned} \tag{A.40}
$$

To avoid duplicating degrees of freedom, in practice we learn filters with fixed unit norm $\mathbf{f} = \frac{\mathbf{B}\tilde{\mathbf{f}}}{\|\mathbf{B}\tilde{\mathbf{f}}\|} = \mathbf{B}\tilde{\mathbf{f}} \cdot \left(\tilde{\mathbf{f}}^\top (\mathbf{B}^\top \mathbf{B}) \tilde{\mathbf{f}}\right)^{-1/2}$. To perform learning for these normalized filters, it is again useful to first derive

$$
\begin{aligned}
\frac{\partial \mathbf{F}\mathbf{x}}{\partial \tilde{\mathbf{f}}} &= \frac{\partial}{\partial \tilde{\mathbf{f}}} [\mathbf{x}]_{\mathcal{C}}^\top \mathbf{B}\tilde{\mathbf{f}} \cdot \left(\tilde{\mathbf{f}}^\top (\mathbf{B}^\top \mathbf{B}) \tilde{\mathbf{f}}\right)^{-1/2} \\
&= [\mathbf{x}]_{\mathcal{C}}^\top \mathbf{B} \cdot \left(\left(\tilde{\mathbf{f}}^\top (\mathbf{B}^\top \mathbf{B}) \tilde{\mathbf{f}}\right)^{-1/2} \cdot \mathbf{I} - \tilde{\mathbf{f}} \cdot \frac{1}{2} \left(\tilde{\mathbf{f}}^\top (\mathbf{B}^\top \mathbf{B}) \tilde{\mathbf{f}}\right)^{-3/2} \cdot 2\tilde{\mathbf{f}}^\top \mathbf{B}^\top \mathbf{B} \right) \\
&= \|\mathbf{B}\tilde{\mathbf{f}}\|^{-1} \cdot \left([\mathbf{x}]_{\mathcal{C}}^\top \mathbf{B} - [\mathbf{x}]_{\mathcal{C}}^\top \mathbf{B}\tilde{\mathbf{f}} \cdot \|\mathbf{B}\tilde{\mathbf{f}}\|^{-1} \|\mathbf{B}\tilde{\mathbf{f}}\|^{-1} \cdot \tilde{\mathbf{f}}^\top \mathbf{B}^\top \mathbf{B} \right) \\
&= \|\mathbf{B}\tilde{\mathbf{f}}\|^{-1} \cdot \left([\mathbf{x}]_{\mathcal{C}}^\top \mathbf{B} - [\mathbf{x}]_{\mathcal{C}}^\top \mathbf{f} \cdot \mathbf{f}^\top \mathbf{B} \right) \\
&= \left([\mathbf{x}]_{\mathcal{C}}^\top - \mathbf{F}\mathbf{x} \cdot \mathbf{f}^\top \right) \frac{\mathbf{B}}{\|\mathbf{B}\tilde{\mathbf{f}}\|}
\end{aligned} \tag{A.41}
$$

to then derive the gradient of the loss w.r.t. the filter coefficients

$$
\begin{aligned}
\frac{\partial \ell(\hat{\mathbf{x}}_t; \mathbf{x}_{\text{gt}})}{\partial \tilde{\mathbf{f}}_{ti}} &= \Big[[\hat{\mathbf{c}}_t]_{\mathcal{C}} \left(f_{\pi_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) - \mathbf{F}_{ti}\hat{\mathbf{x}}_t^{\mathrm{B}} \right) \\
&\qquad + [\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}]_{\mathcal{C}} f_{\pi_{ti}}'(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) \mathbf{F}_{ti}\hat{\mathbf{c}}_t - [\hat{\mathbf{x}}_t^{\mathrm{B}}]_{\mathcal{C}} \mathbf{F}_{ti}\hat{\mathbf{c}}_t \Big]^\top \frac{\mathbf{B}}{\|\mathbf{B}\tilde{\mathbf{f}}_{ti}\|} \\
&\qquad - \Big[(\mathbf{F}_{ti}\hat{\mathbf{c}}_t)^\top \left(f_{\pi_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) - \mathbf{F}_{ti}\hat{\mathbf{x}}_t^{\mathrm{B}} \right) \\
&\qquad\quad + (\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}})^\top f_{\pi_{ti}}'(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) \mathbf{F}_{ti}\hat{\mathbf{c}}_t - (\mathbf{F}_{ti}\hat{\mathbf{x}}_t^{\mathrm{B}})^\top \mathbf{F}_{ti}\hat{\mathbf{c}}_t \Big] \frac{\mathbf{f}^\top \mathbf{B}}{\|\mathbf{B}\tilde{\mathbf{f}}_{ti}\|} \\
&= \Big[[\hat{\mathbf{c}}_t]_{\mathcal{C}} \left(f_{\pi_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) - \mathbf{F}_{ti}\hat{\mathbf{x}}_t^{\mathrm{B}} \right) \\
&\qquad + [\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}]_{\mathcal{C}} f_{\pi_{ti}}'(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) \mathbf{F}_{ti}\hat{\mathbf{c}}_t - [\hat{\mathbf{x}}_t^{\mathrm{B}}]_{\mathcal{C}} \mathbf{F}_{ti}\hat{\mathbf{c}}_t \Big]^\top \frac{\mathbf{B}}{\|\mathbf{B}\tilde{\mathbf{f}}_{ti}\|} \\
&\qquad - (\mathbf{F}_{ti}\hat{\mathbf{c}}_t)^\top \Big[f_{\pi_{ti}}(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) - 2\mathbf{F}_{ti}\hat{\mathbf{x}}_t^{\mathrm{B}} \\
&\qquad\quad + f_{\pi_{ti}}'(\mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}}) \mathbf{F}_{ti}\hat{\mathbf{x}}_{t-1}^{\mathrm{P}} \Big] \frac{\mathbf{f}^\top \mathbf{B}}{\|\mathbf{B}\tilde{\mathbf{f}}_{ti}\|},
\end{aligned} \tag{A.42}
$$

where we can re-use most of our previous derivation (Eq. A.40).

BIBLIOGRAPHY

Timo Ahonen, Jiří Matas, Chu He, and Matti Pietikäinen. Rotation invariant image description with local binary pattern histogram Fourier features. In Arnt-Børre Salberg, Jon Yngve Hardeberg, and Robert Jenssen, editors, *Image Analysis*, volume 5575 of *Lecture Notes in Computer Science*, pages 61–70. Springer, 2009.

Marc Allain, Jérôme Idier, and Yves Goussard. On global and local convergence of half-quadratic algorithms. *IEEE Transactions on Image Processing*, 15(5):1130–1142, May 2006.

Josue Anaya and Adrian Barbu. RENOIR - A benchmark dataset for real noise reduction evaluation. *arXiv:1409.8230*, 2014.

D. F. Andrews and C. L. Mallows. Scale mixtures of normal distributions. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, 36(1):99–102, 1974.

Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1):5–43, January 2003.

Anestis Antoniadis and Jianqing Fan. Regularization of wavelet approximations. *Journal of the American Statistical Association*, 96(455): 939–967, 2001.

Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898–916, 2011.

S Derin Babacan, Rafael Molina, Minh N Do, and Aggelos K Katsaggelos. Bayesian blind deconvolution with general sparse image priors. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Proceedings of the 12th European Conference on Computer Vision*, volume 7577 of *Lecture Notes in Computer Science*, pages 341–355. Springer, 2012.

Hicham Badri, Hussein Yahia, and Driss Aboutajdine. Fast edge-aware processing via first order proximal approximation. *IEEE Transactions on Visualization and Computer Graphics*, PP(99):14, January 2015.

Adrian Barbu. Training an active random field for real-time image denoising. *IEEE Transactions on Image Processing*, 18(11):2451–2462, November 2009.

Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 28(3):24:1–24:11, August 2009.

Connelly Barnes, Eli Shechtman, Dan B. Goldman, and Adam Finkelstein. The generalized PatchMatch correspondence algorithm. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Proceedings of the 11th European Conference on Computer Vision*, volume 6313 of *Lecture Notes in Computer Science*, pages 29–43. Springer, September 2010.

Bryce E. Bayer. Color imaging array. US Patent 3 971 065, July 1976.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

Moshe Ben-Ezra and Shree K. Nayar. Motion-based motion deblurring. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):689–698, June 2004.

James O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer New York, 1985.

James Bergstra, Aaron Courville, and Yoshua Bengio. The statistical inefficiency of sparse coding for images (or, one Gabor to rule them all). Technical Report 1109.6638v2, arXiv, 2011.

Julian Besag. Spatial interaction and the statistical analysis of lattices. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, 36(2):192–236, 1974.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

Michael J. Black and Anand Rangarajan. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *International Journal of Computer Vision*, 19(1):57–91, July 1996.

Andrew Blake and Andrew Zisserman. *Visual Reconstruction*. MIT Press, 1987.

Michael Bleyer, Christoph Rhemann, and Carsten Rother. PatchMatch stereo - stereo matching with slanted support windows. In *Proceedings of the British Machine Vision Conference*, Dundee, UK, August–September 2011.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Tomás Brandão and Maria Paula Queluz. No-reference image quality assessment based on DCT domain statistics. *Signal Processing*, 88 (4):822–833, 2008.

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, October 2001.

Matthew Brown, Gang Hua, and Simon Winder. Discriminative learning of local image descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1), 2011.

Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. A non-local algorithm for image denoising. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 60–65, San Diego, California, June 2005.

Harold C. Burger, Christian J. Schuler, and Stefan Harmeling. Image denoising: Can plain neural networks compete with BM3D? In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Providence, Rhode Island, June 2012.

Harold C. Burger, Christian Schuler, and Stefan Harmeling. Learning how to combine internal and external denoising methods. In J. Weickert, M. Hein, and B. Schiele, editors, *Proceedings of the German Conference on Pattern Recognition*, volume 8142 of *Lecture Notes in Computer Science*, pages 121–130. Springer, September 2013.

Miguel Á. Carreira-Perpiñán and Geoffrey E. Hinton. On contrastive divergence learning. In *Proceedings of the Eighth International Workshop on Artificial Intelligence and Statistics*, pages 33–40, Barbados, January 2005.

Vladimír Černý. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41–51, 1985.

Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

Frédéric Champagnat and Jérôme Idier. A connection between half-quadratic criteria and EM algorithms. *IEEE Signal Processing Letters*, 11(9):709–712, September 2004.

Raymond H. Chan and Michael K. Ng. Conjugate gradient methods for Toeplitz systems. *SIAM Review*, 38(3):427–482, 1996.

Tony F. Chan and Pep Mulet. On the convergence of the lagged diffusivity fixed point method in total variation image restoration. *SIAM Journal on Numerical Analysis*, 36(2):354–367, 1999.

Damon M. Chandler and Sheila S. Hemami. VSNR: A wavelet-based visual signal-to-noise ratio for natural images. *IEEE Transactions on Image Processing*, 16(9):2284–2298, 2007.

Giannis Chantas, Nikolaos Galatsanos, Aristidis Likas, and Michael Saunders. Variational Bayesian image restoration based on a product of t-distributions image prior. *IEEE Transactions on Image Processing*, 17(10):1795–1805, October 2008.

Pierre Charbonnier, Laure Blanc-Féraud, Gilles Aubert, and Michel Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings of the IEEE International Conference on Image Processing*, volume 2, pages 168–172, Austin, Texas, November 1994.

Pierre Charbonnier, Laure Blanc-Feéraud, Gilles Aubert, and Michel Barlaud. Deterministic edge-preserving regularization in computed imaging. *IEEE Transactions on Image Processing*, 6(2):298–311, February 1997.

Priyam Chatterjee, Neel Joshi, Sing Bing Kang, and Yasuyuki Matsushita. Noise suppression in low-light images through joint denoising and demosaicing. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Colorado Springs, Colorado, June 2011.

Yunjin Chen, Thomas Pock, René Ranftl, and Horst Bischof. Revisiting loss-specific training of filter-based MRFs for image restoration. In J. Weickert, M. Hein, and B. Schiele, editors, *Proceedings of the German Conference on Pattern Recognition*, volume 8142 of *Lecture Notes in Computer Science*, pages 271–281. Springer, 2013.

Yunjin Chen, Wei Yu, and Thomas Pock. On learning optimized reaction diffusion processes for effective image restoration. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Boston, Massachusetts, June 2015.

Sunghyun Cho and Seungyong Lee. Fast motion deblurring. *ACM Transactions on Graphics*, 28(5):145:1–145:8, December 2009.

Sunghyun Cho, Jue Wang, and Seungyong Lee. Handling outliers in non-blind image deconvolution. In *Proceedings of the Thirteenth IEEE International Conference on Computer Vision*, Barcelona, Spain, November 2011.

Taeg Sang Cho, Neel Joshi, C. Lawrence Zitnick, Sing Bing Kang, Richard Szeliski, and William T. Freeman. A content-aware image prior. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 169–176, San Francisco, California, June 2010.

Taeg Sang Cho, C. Lawrence Zitnick, Neel Joshi, Sing Bing Kang, Richard Szeliski, and William T. Freeman. Image restoration by matching gradient distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):683–694, April 2012.

Benoît Colson, Patrice Marcotte, and Gilles Savard. An overview of bilevel optimization. *Annals of Operations Research*, 153(1):235–256, 2007.

Patrick L. Combettes and Jean-Christophe Pesquet. Proximal splitting methods in signal processing. In Heinz H. Bauschke, Regina S. Burachik, Patrick L. Combettes, Veit Elser, D. Russell Luke, and Henry Wolkowicz, editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, Springer Optimization and Its Applications, pages 185–212. Springer, 2011.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

Aaron Courville, James Bergstra, and Yoshua Bengio. A spike and slab restricted Boltzmann machine. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, Ft. Lauderdale, Florida, April 2011.

Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Color image denoising via sparse 3D collaborative filtering with grouping constraint in luminance-chrominance space. In *Proceedings of the IEEE International Conference on Image Processing*, San Antonio, Texas, September 2007a.

Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8): 2080–2095, August 2007b.

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 886–893, San Diego, California, June 2005.

Aram Danielyan, Vladimir Katkovnik, and Karen Egiazarian. BM3D frames and variational image deblurring. *IEEE Transactions on Image Processing*, 21(4):1715–1728, 2012.

Anirban DasGupta. Distributions which are Gaussian convolutions. In Shanti S. Gupta and James O. Berger, editors, *Statistical Decision Theory and Related Topics*, volume V, pages 391–400. Springer, 1994.

Antonio De Stefano, Paul R. White, and William B. Collis. Training methods for image noise level estimation on wavelet components.

EURASIP Journal on Applied Signal Processing (Special issue on Non-Linear Signal and Image Processing–Part II), 2004(16):2400–2407, January 2004.

Frank Dellaert. The expectation maximization algorithm. Technical Report GIT-GVU-02-20, College of Computing, Georgia Institute of Technology, February 2002.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, 39(1):1–38, 1977.

Piotr Dollár. Piotr's Computer Vision Matlab Toolbox (PMT). `https://github.com/pdollar/toolbox`.

Justin Domke. Generic methods for optimization-based modeling. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, pages 318–326, La Palma, Canary Islands, April 2012.

Justin Domke, Alap Karapurkar, and Yiannis Aloimonos. Who killed the directed model? In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, June 2008.

David L. Donoho. Denoising by soft-thresholding. *IEEE Transactions on Information Theory*, 41(3):613–627, May 1995.

David L. Donoho and Iain M. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455, 1994.

DxO Image Science. Optics Pro 9: PRIME. `http://download-center.dxo.com/Press/opticspro/v9/pr/PR-DxO-Optics-Pro-9-EN.pdf`, October 2013.

David J. Earl and Michael W. Deem. Parallel tempering: Theory, applications, and new perspectives. *Physical Chemistry Chemical Physics*, 7(23):3910–3916, 2005.

Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image Processing*, 15(12):3736–3745, December 2006.

Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. *The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results*. 2010.

Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

Pedro F. Felzenszwalb, Ross. B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

Rob Fergus, Barun Singh, Aaron Hertzmann, Sam T. Roweis, and William T. Freeman. Removing camera shake from a single photograph. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 25(3):787–794, July-August 2006.

Wolfgang Förstner. Reliability analysis of parameter estimation in linear models with applications to mensuration problems in computer vision. *Computer Vision, Graphics, and Image Processing*, 40(3): 273–310, 1987.

William T. Freeman and Edward H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906, September 1991.

Brendan J. Frey and Nebojsa Jojic. Transformed component analysis: Joint estimation of spatial transformations and image components. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Fort Collins, Colorado, June 1999.

Björn Fröhlich, Erik Rodner, and Joachim Denzler. As time goes by—anytime semantic segmentation with iterative context forests. In A. Pinz, T. Pock, H. Bischof, and F. Leberl, editors, *Pattern Recognition, Proceedings of the 34th DAGM-Symposium*, volume 7476 of *Lecture Notes in Computer Science*. Springer, 2012.

Qi Gao and Stefan Roth. How well do filter-based MRFs model natural images? In A. Pinz, T. Pock, H. Bischof, and F. Leberl, editors, *Pattern Recognition, Proceedings of the 34th DAGM-Symposium*, volume 7476 of *Lecture Notes in Computer Science*, pages 62–72. Springer, 2012.

Peter V. Gehler, Carsten Rother, Martin Kiefel, Lumin Zhang, and Bernhard Schölkopf. Recovering intrinsic images with a global sparsity prior on reflectance. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24, pages 765–773, 2011.

Davi Geiger and Frederico Girosi. Parallel and deterministic algorithms from MRF's: Surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):401–412, May 1991.

Alan E. Gelfand and Adrian F. M. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990.

Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC, 2004.

Donald Geman and George Reynolds. Constrained restoration and the recovery of discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(3):367–383, March 1992.

Donald Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing*, 4(7):932–946, July 1995.

Stuart Geman and Donald Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, November 1984.

Clément Gilavert, Saïd Moussaoui, and Jérôme Idier. Efficient Gaussian sampling for solving large-scale inverse problems using MCMC. *IEEE Transactions on Signal Processing*, 63(1):70–80, January 2015.

Tilmann Gneiting. Normal scale mixtures and dual probability densities. *Journal of Statistical Computation and Simulation*, 59(4):375–384, 1997.

Tom Goldstein and Stanley Osher. The split Bregman method for L1-regularized problems. *SIAM Journal on Imaging Sciences*, 2(2): 323–343, 2009.

Gene H. Golub and Charles F. van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

Robert M. Gray. Toeplitz and circulant matrices: A review. *Foundations and Trends in Communications and Information Theory*, 2(3): 155–239, 2006.

Michael U. Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(1):307–361, February 2012.

Jacques Hadamard. *Lectures on the Cauchy Problem in Linear Partial Differential Equations*. Yale University Press, New Haven, Connecticut, 1923.

William W. Hager. Updating the inverse of a matrix. *SIAM Review*, 31 (2):221–239, 1989.

John M. Hammersley and Peter Clifford. Markov fields on finite graphs and lattices. Unpublished manuscript, 1971.

W. Keith Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

Geremy Heitz and Daphne Koller. Learning spatial context: Using stuff to find things. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Proceedings of the Tenth European Conference on Computer Vision*, volume 5302 of *Lecture Notes in Computer Science*. Springer, 2008.

Yacov Hel-Or and Doron Shaked. A discriminative approach for wavelet denoising. *IEEE Transactions on Image Processing*, 17(4):443–457, 2008.

Magnus Rudolph Hestenes and Eduard Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6), December 1952.

Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, August 2002.

Geoffrey E. Hinton and Ruslan R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 2006.

Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1–3):185–203, August 1981.

Michael Hornáček, Frederic Besse, Jan Kautz, Andrew Fitzgibbon, and Carsten Rother. Highly overparameterized optical flow using PatchMatch belief propagation. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Proceedings of the 13th European Conference on Computer Vision*, volume 8691 of *Lecture Notes in Computer Science*, pages 220–234. Springer, September 2014.

Zhe Hu, Sunghyun Cho, Jue Wang, and Ming-Hsuan Yang. Deblurring low-light images with light streaks. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Columbus, Ohio, June 2014.

Jinggang Huang. *Statistics of Natural Images and Models*. PhD thesis, Brown University, 2000.

Sabine Husse, Yves Goussard, and Jérôme Idier. Extended forms of Geman & Yang algorithm: application to MRI reconstruction. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume III, pages 513–516, Montreal, Quebec, Canada, May 2004.

Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–708, April 2005.

Aapo Hyvärinen and Erkki Oja. Independent component analysis: Algorithms and applications. *Neural Networks*, 13(4–5):411–430, June 2000.

Aapo Hyvärinen, Jarmo Hurri, and Patrick O. Hoyer. *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision*. Springer, 2009.

Jérôme Idier. Convex half-quadratic criteria and interacting auxiliary variables for image restoration. *IEEE Transactions on Image Processing*, 10(7):1001–1009, July 2001.

Alexander T. Ihler, Erik B. Sudderth, William T. Freeman, and Alan S. Willsky. Efficient multiscale sampling from products of Gaussian mixtures. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, page ?, 2004.

A. Jain. Fast inversion of banded Toeplitz matrices by circular decompositions. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 26(2):121–126, 1978.

Viren Jain and H. Sebastian Seung. Natural image denoising with convolutional networks. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21, pages 769–776, 2009.

Varun Jampani, Sebastian Nowozin, Matthew Loper, and Peter V. Gehler. The informed sampler: A discriminative approach to Bayesian inference in generative computer vision models. *Computer Vision and Image Understanding, Special Issue on Generative Models in Computer Vision and Medical Imaging*, 136:32–44, July 2015.

Jeremy Jancsary, Sebastian Nowozin, and Carsten Rother. Loss-specific training of non-parametric image restoration models: A new state of the art. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Proceedings of the 12th European Conference on Computer Vision*, volume 7578 of *Lecture Notes in Computer Science*. Springer, 2012a.

Jeremy Jancsary, Sebastian Nowozin, Toby Sharp, and Carsten Rother. Regression tree fields – an efficient, non-parametric approach to image labeling problems. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Providence, Rhode Island, June 2012b.

Johan L. W. V. Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, 30(1):175–193, 1906.

Ian T. Jolliffe. *Principal Component Analysis*. Springer, New York, New York, 2nd edition, 2002.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, November 1999.

Neel Joshi, Richard Szeliski, and David J. Kriegman. PSF estimation using sharp edge prediction. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, June 2008.

Neel Joshi, Sing Bing Kang, C. Lawrence Zitnick, and Richard Szeliski. Image deblurring using inertial measurement sensors. *ACM Transactions on Graphics*, 29(4):30:1–30:9, July 2010.

Aggelos K. Katsaggelos. *Digital Image Restoration*. Springer Berlin Heidelberg, 2012.

Koray Kavukcuoglu, Marc'Aurelio Ranzato, Rob Fergus, and Yann LeCun. Learning invariant features through topographic filter maps. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami, Florida, June 2009.

Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

Jyri J. Kivinen and Christopher K. I. Williams. Transformation equivariant Boltzmann machines. In *Proceedings of the 20th International Conference on Artificial Neural Networks*, Espoo, Finland, June 2011.

Rolf Köhler, Michael Hirsch, Betty Mohler, Bernhard Schölkopf, and Stefan Harmeling. Recording and playback of camera shake: Benchmarking blind deconvolution with a real-world database. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Proceedings of the 12th European Conference on Computer Vision*, volume 7578 of *Lecture Notes in Computer Science*. Springer, 2012.

Iasonas Kokkinos and Alan Yuille. Scale invariance without scale selection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, June 2008.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

Kai Krajsek and Rudolf Mester. A maximum likelihood estimator for choosing the regularization parameters in global optical flow methods. In *Proceedings of the IEEE International Conference on Image Processing*, Atlanta, Georgia, October 2006.

Dilip Krishnan and Rob Fergus. Fast image deconvolution using hyper-Laplacian priors. In *Advances in Neural Information Processing Systems*, volume 22, pages 1033–1041, 2009.

Dilip Krishnan and Richard Szeliski. Multigrid and multilevel preconditioners for computational photography. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia)*, 30(6):177:1–177:10, December 2011.

Dilip Krishnan, Raanan Fattal, and Richard Szeliski. Efficient preconditioning of Laplacian matrices for computer graphics. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 32(4):142:1–142:15, July 2013.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105, 2012.

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, Williamstown, Massachusetts, June-July 2001.

Edwin H. Land and John J. McCann. Lightness and retinex theory. *Journal of the Optical Society of America*, 61(1):1–11, January 1971.

Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, Oregon, June 2007.

Hugo Larochelle, Dumitru Erhan, and Pascal Vincent. Deep learning using robust interdependent codes. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, Clearwater Beach, Florida, April 2009.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Semi-local affine parts for object recognition. In *Proceedings of the British Machine Vision Conference*, Kingston, UK, September 2004.

Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, New York, New York, June 2006.

Nicolas Le Roux, Nicolas Heess, Jamie Shotton, and John Winn. Learning a generative model of images by factoring appearance and shape. *Neural Computation*, 23(3):593–650, March 2011.

Yann LeCun and Corinna Cortes. The MNIST database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`.

Yann LeCun, Koray Kavukcuoglu, and Clément Farabet. Convolutional networks and applications in vision. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, pages 253–256, May 2010.

Honglak Lee, Chaitanya Ekanadham, and Andrew Y. Ng. Sparse deep belief net model for visual area V2. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20, 2008.

Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, June 2009.

Karel Lenc and Andrea Vedaldi. Understanding image representations by measuring their equivariance and equivalence. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Boston, Massachusetts, June 2015.

Effi Levi. Using natural image priors – Maximizing or sampling? Master's thesis, The Hebrew University of Jerusalem, 2009.

Anat Levin, Rob Fergus, Frédo Durand, and William T. Freeman. Image and depth from a conventional camera with a coded aperture. *ACM Transactions on Graphics*, 26(3):70:1–70:9, July 2007.

Anat Levin, Yair Weiss, Fredo Durand, and William T. Freeman. Understanding and evaluating blind deconvolution algorithms. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami, Florida, June 2009.

Anat Levin, Yair Weiss, Fredo Durand, and William T. Freeman. Efficient marginal likelihood optimization in blind deconvolution. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Colorado Springs, Colorado, June 2011.

Stan Z. Li. *Markov Random Field Modeling in Image Analysis*. Springer, 2nd edition, 2001.

Percy Liang and Michael I. Jordan. An asymptotic analysis of generative, discriminative, and pseudolikelihood estimators. In *Proceedings of the 25th International Conference on Machine Learning*, pages 584–591, Helsinki, Finnland, July 2008.

Ce Liu, William T. Freeman, Richard Szeliski, and Sing Bing Kang. Noise estimation from a single image. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 901–908, New York, New York, June 2006.

Kun Liu, Henrik Skibbe, Thorsten Schmidt, Thomas Blein, Klaus Palme, Thomas Brox, and Olaf Ronneberger. Rotation-invariant HOG descriptors using Fourier analysis in polar and spherical coordinates. *International Journal of Computer Vision*, 106(3):342–364, 2014.

David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004.

Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 1981 DARPA Image Understanding Workshop*, pages 121–130, April 1981.

Leon B. Lucy. An iterative technique for the rectification of observed distributions. *The Astronomical Journal*, 79(6), 1974.

Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Non-local sparse models for image restoration. In *Proceedings of the Twelfth IEEE International Conference on Computer Vision*, Kyoto, Japan, October 2009.

Stéphane Mallat. *A Wavelet Tour of Signal Processing: The Sparse Way*. Academic Press, 2009.

David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings of the Eighth IEEE International Conference on Computer Vision*, volume 2, pages 416–423, Vancouver, British Columbia, Canada, July 2001.

Xing Mei, Weiming Dong, Bao-Gang Hu, and Siwei Lyu. UniHIST: A unified framework for image restoration with marginal histogram constraints. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3753–3761, Boston, Massachusetts, June 2015.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092, 1953.

Thomas P. Minka. Expectation-maximization as lower bound maximization. 1998.

James Miskin and David J. C. MacKay. Ensemble learning for blind image separation and deconvolution. *Adv. in Ind. Comp. Analysis*, 2000.

Inbar Mosseri, Maria Zontak, and Michal Irani. Combining the power of internal and external denoising. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, editors, *Proceedings of the IEEE International Conference on Computational Photography*, pages 1–9, April 2013.

Radford M. Neal. Sampling from multimodal distributions using tempered transitions. *Statistics and Computing*, 6(4):353–366, 1996.

Mila Nikolova and Raymond H. Chan. The equivalence of half-quadratic minimization and the gradient linearization iteration. *IEEE Transactions on Image Processing*, 16(6):1623–1627, 2007.

Mila Nikolova and Michael K. Ng. Analysis of half-quadratic minimization methods for signal and image recovery. *SIAM Journal on Scientific computing*, 27(3):937–966, 2005.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 1999.

Mohammad Norouzi, Mani Ranjbar, and Greg Mori. Stacks of convolutional restricted Boltzmann machines for shift-invariant feature learning. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami, Florida, June 2009.

Kenji Okuma, Ali Taleghani, O. De Freitas, James J. Little, and David G. Lowe. A boosted particle filter: Multitarget detection and tracking. In *Proceedings of the Eighth European Conference on Computer Vision*, volume 3021 of *Lecture Notes in Computer Science*, pages 28–39. Springer, 2004. ISBN 3-540-21984-6.

François Orieux, Olivier Féron, and Jean-François Giovannelli. Sampling high-dimensional Gaussian distributions for general linear inverse problems. *IEEE Signal Processing Letters*, 19(5):251–254, May 2012.

Jason Palmer, Kenneth Kreutz-Delgado, Bhaskar D. Rao, and David P. Wipf. Variational EM algorithms for non-Gaussian latent variable models. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18, pages 1059–1066, 2006.

George Papandreou and Alan L. Yuille. Gaussian sampling by local perturbations. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 23, pages 1858–1866, 2010.

Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2013.

Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, July 1990.

Patrick Pletscher, Sebastian Nowozin, Pushmeet Kohli, and Carsten Rother. Putting MAP back on the map. In Rudolf Mester and Michael Felsberg, editors, *Pattern Recognition, Proceedings of the 33rd DAGM-Symposium*, volume 6835 of *Lecture Notes in Computer Science*, pages 111–121. Springer, August 2011.

Nicholas G. Polson and James G. Scott. Mixtures, envelopes and hierarchical duality. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, 78(4):701–727, 2016.

Javier Portilla, Vasily Strela, Martin J. Wainwright, and Eero P. Simoncelli. Image denoising using scale mixtures of Gaussians in the wavelet domain. *IEEE Transactions on Image Processing*, 12(11): 1338–1351, November 2003.

Howard Raiffa and Robert Schlaifer. *Applied Statistical Decision Theory*. Harvard Business School, Boston, 1961.

Donald B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6):843–854, December 1979.

William Hadley Richardson. Bayesian-based iterative method of image restoration. *Journal of the Optical Society of America*, 62(1):55–59, January 1972.

R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

Isabel Rodrigues, João Sanches, and José Bioucas-Dias. Denoising of medical images corrupted by Poisson noise. In *Proceedings of the IEEE International Conference on Image Processing*, pages 1756–1759, San Diego, California, October 2008.

Stephane Ross, Daniel Munoz, Martial Hebert, and J. Andrew Bagnell. Learning message-passing inference machines for structured prediction. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Colorado Springs, Colorado, June 2011.

Stefan Roth. *High-Order Markov Random Fields for Low-Level Vision*. Ph.D. dissertation, Brown University, Department of Computer Science, Providence, Rhode Island, May 2007.

Stefan Roth and Michael J. Black. Fields of experts. *International Journal of Computer Vision*, 82(2):205–229, April 2009.

Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. "GrabCut": Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 23(3):309–314, August 2004.

Donald B. Rubin. Iteratively reweighted least squares. In *Encyclopedia of Statistical Sciences*, volume 4, pages 272–275. Wiley, 1983.

Daniel L. Ruderman. The statistics of natural images. *Network: Computation in Neural Systems*, 5(4):517–548, November 1994.

Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, November 1992.

Håvard Rue. Fast sampling of Gaussian Markov random fields. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63 (2):325–338, 2001.

Håvard Rue and Leonhard Held. *Gaussian Markov Random Fields. Theory and Applications.* Chapman & Hall / CRC, 2005.

Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, second edition, 2003.

Kegan G. G. Samuel and Marshall F. Tappen. Learning optimized MAP estimates in continuously-valued MRF models. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Miami, Florida, June 2009.

Kevin Schelten and Stefan Roth. Connecting non-quadratic variational models and MRFs. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2641–2648, Colorado Springs, Colorado, June 2011.

Kevin Schelten and Stefan Roth. Mean field for continuous high-order MRFs. In A. Pinz, T. Pock, H. Bischof, and F. Leberl, editors, *Pattern Recognition, Proceedings of the 34th DAGM-Symposium*, volume 7476 of *Lecture Notes in Computer Science*, pages 52–61. Springer, 2012.

Kevin Schelten, Sebastian Nowozin, Jeremy Jancsary, Carsten Rother, and Stefan Roth. Interleaved regression tree field cascades for blind image deconvolution. In *IEEE Winter Conference on Applications of Computer Vision*, pages 494–501, Waikoloa Beach, HI, January 2015.

Mark Schmidt. minFunc: unconstrained differentiable multivariate optimization in Matlab. `http://www.cs.ubc.ca/~schmidtm/Software/minFunc.html`, 2005.

Uwe Schmidt. Learning and evaluating Markov random fields for natural images. M.Sc. thesis, TU Darmstadt, Germany, February 2010.

Uwe Schmidt and Stefan Roth. Learning rotation-aware features: From invariant priors to equivariant descriptors. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2050–2057, Providence, Rhode Island, June 2012.

Uwe Schmidt and Stefan Roth. Shrinkage fields for effective image restoration. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2774–2781, Columbus, Ohio, June 2014.

Uwe Schmidt, Qi Gao, and Stefan Roth. A generative perspective on MRFs in low-level vision. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1751–1758, San Francisco, California, June 2010.

Uwe Schmidt, Kevin Schelten, and Stefan Roth. Bayesian deblurring with integrated noise estimation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2625–2632, Colorado Springs, Colorado, June 2011.

Uwe Schmidt, Carsten Rother, Sebastian Nowozin, Jeremy Jancsary, and Stefan Roth. Discriminative non-blind deblurring. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 604–611, Portland, Oregon, June 2013.

Uwe Schmidt, Jeremy Jancsary, Sebastian Nowozin, Stefan Roth, and Carsten Rother. Cascades of regression tree fields for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(4):677–689, April 2016.

Christoph Schnörr, Rainer Sprengel, and Bernd Neumann. A variational approach to the design of early vision algorithms. *Computing Supplement*, 11:149–165, 1996.

Christian J. Schuler, Harold Christopher Burger, Stefan Harmeling, and Bernhard Schölkopf. A machine learning approach for non-blind image deconvolution. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1067–1074, Portland, Oregon, June 2013.

Qi Shan, Jiaya Jia, and Aseem Agarwala. High-quality motion deblurring from a single image. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 27(3), 2008.

Hamid R. Sheikh and Alan C. Bovik. Image information and visual quality. *IEEE Transactions on Image Processing*, 15(2):430–444, 2006.

Eero P. Simoncelli. Bayesian denoising of visual images in the wavelet domain. In P. Müller and B. Vidakovic, editors, *Bayesian Inference in Wavelet Based Models*, volume 141 of *Lecture Notes in Statistics*, pages 292–308. Springer, 1999.

Jascha Sohl-Dickstein, Peter B. Battaglino, and Michael R. DeWeese. New method for parameter estimation in probabilistic models: Minimum probability flow. *Physical Review Letters*, 107(22):220601, November 2011.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32th International Conference on Machine Learning*, Lille, France, July 2015.

Kihyuk Sohn and Honglak Lee. Learning invariant representations with local transformations. In *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland, June 2012.

Deqing Sun, Stefan Roth, and Michael J. Black. Secrets of optical flow estimation and their principles. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2432–2439, San Francisco, California, June 2010.

Jian Sun and Marshall F. Tappen. Learning non-local range Markov random field for image restoration. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2745–2752, Colorado Springs, Colorado, June 2011.

Jian Sun, Nan-Ning Zhen, and Heung-Yeung Shum. Stereo matching using belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(7):787–800, July 2003.

Libin Sun, Sunghyun Cho, Jue Wang, and James Hays. Good image priors for non-blind deconvolution. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Proceedings of the 13th European Conference on Computer Vision*, volume 8692 of *Lecture Notes in Computer Science*, pages 231–246. Springer, September 2014.

Kevin Swersky, Bo Chen, Ben Marlin, and Nando de Freitas. A tutorial on stochastic approximation algorithms for training restricted Boltzmann machines and deep belief nets. In *Information Theory and Applications Workshop*, pages 1–10, January 2010.

Richard Szeliski. Bayesian modeling of uncertainty in low-level vision. *International Journal of Computer Vision*, 5(3):271–301, December 1990.

Yu-Wing Tai and Stephen Lin. Motion-aware noise filtering for deblurring of noisy and blurry images. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Providence, Rhode Island, June 2012.

Yu-Wing Tai, Hao Du, Michael S. Brown, and Stephen Lin. Image/video deblurring using a hybrid camera. In *Proceedings of* *the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, June 2008.

Gabriel Takacs, Vijay Chandrasekhar, Huizhong Chen, David Chen, Sam Tsai, Radek Grzeszczuk, and Bernd Girod. Permutable descriptors for orientation-invariant image matching. In Andrew G. Tescher, editor, *Proceedings of SPIE (Applications of Digital Image Processing XXXIII)*, volume 7798, San Diego, California, 2010.

Marshall Tappen, Ce Liu, Edward H. Adelson, and William T. Freeman. Learning Gaussian conditional random fields for low-level vision. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Minneapolis, Minnesota, June 2007.

Marshall F. Tappen. Utilizing variational optimization to learn Markov random fields. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Minneapolis, Minnesota, June 2007.

Maja Temerinac-Ott, Olaf Ronneberger, Peter Ochs, Wolfgang Driever, Thomas Brox, and Hans Burkhardt. Multiview deblurring for 3-D images from light-sheet-based fluorescence microscopy. *IEEE Transactions on Image Processing*, 21(4):1863–1873, 2012.

Lucas Theis, Reshad Hosseini, and Matthias Bethge. Mixtures of conditional Gaussian scale mixtures applied to multiscale image representations. *PLoS ONE*, 7(7):e39857, 2012.

Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th International Conference on Machine Learning*, Helsinki, Finnland, July 2008.

Andrey N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. *Soviet Math. Dokl.*, 4:1035–1038, 1963.

Zhuowen Tu. Auto-context and its application to high-level vision tasks. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, June 2008.

Vladimir Vapnik and Alexey Chervonenkis. *Theory of pattern recognition (in Russian)*. Nauka, Moscow, 1974.

Andrea Vedaldi, Matthew Blaschko, and Andrew Zisserman. Learning equivariant structured output SVM regressors. In *Proceedings of the Thirteenth IEEE International Conference on Computer Vision*, Barcelona, Spain, November 2011.

Jean-Philippe Vert, Koji Tsuda, and Bernhard Schölkopf. *Kernel Methods in Computational Biology*, chapter A Primer on Kernel Methods, pages 35–70. MIT Press, 2004.

Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Kauai, Hawaii, December 2001.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.

Martin J. Wainwright and Eero P. Simoncelli. Scale mixtures of Gaussians and the statistics of natural images. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12, pages 855–861, 2000.

Gregory K. Wallace. The JPEG still picture compression standard. *Communications of the ACM*, 34(4):30–44, April 1991.

Xiaoyan Wang, Chunping Hou, Liangzhou Pu, and Yonghong Hou. A depth estimating method from a single image using FoE CRF. *Multimedia Tools and Applications*, 2014.

Yilun Wang, Junfeng Yang, Wotao Yin, and Yin Zhang. A new alternating minimization algorithm for Total Variation image reconstruction. *SIAM Journal on Imaging Sciences*, 1(3):248–272, 2008.

Zhou Wang, Hamid R. Sheikh, and Alan C. Bovik. No-reference perceptual quality assessment of JPEG compressed images. In *Proceedings of the IEEE International Conference on Image Processing*, Rochester, New York, September 2002.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, April 2004.

Joachim Weickert. A review of nonlinear diffusion filtering. In *Proceedings of Scale-Space Theory in Computer Vision*, volume 1252 of *Lecture Notes in Computer Science*, pages 3–28, Berlin, Germany, 1997. Springer.

Endre Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tôhoku Mathematical Journal*, 43: 355–386, 1937.

Max Welling, Geoffrey E. Hinton, and Simon Osindero. Learning sparse topographic representations with products of Student-t distributions. In S. Becker, S. Thrun, and K. Obermayer, editors, *Ad-*

vances in Neural Information Processing Systems, volume 15, pages 1359–1366, 2003.

Halbert White. Maximum likelihood estimation of misspecified models. *Econometrica*, 50(1):1–25, January 1982.

Oliver Whyte, Josef Sivic, Andrew Zisserman, and Jean Ponce. Non-uniform deblurring for shaken images. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 491–498, San Francisco, California, June 2010.

Oliver Whyte, Josef Sivic, and Andrew Zisserman. Deblurring shaken and partially saturated images. *International Journal of Computer Vision*, 110(2):185–201, 2014.

David V. Widder. *The Laplace Transform*. Princeton University Press, 1946.

Norbert Wiener. *The Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. John Wiley & Sons, Inc., New York, N. Y., 1949.

David Wipf and Haichao Zhang. Revisiting Bayesian blind deconvolution. *Journal of Machine Learning Research*, 15:3595–3634, November 2014.

Li Xu and Jiaya Jia. Two-phase kernel estimation for robust motion deblurring. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Proceedings of the 11th European Conference on Computer Vision*, volume 6311 of *Lecture Notes in Computer Science*. Springer, 2010.

Laurent Younes. Parametric inference for imperfectly observed Gibbsian fields. *Probability Theory and Related Fields*, 82(4):625–645, 1989.

Lu Yuan, Jian Sun, Long Quan, and Heung-Yeung Shum. Image deblurring with blurred/noisy image pairs. *ACM Transactions on Graphics*, 26(3), July 2007.

Lu Yuan, Jian Sun, Long Quan, and Heung-Yeung Shum. Progressive inter-scale and intra-scale non-blind image deconvolution. *ACM Transactions on Graphics*, 27(3), 2008.

Haichao Zhang and Yanning Zhang. Bayesian image separation with natural image prior. In *Proceedings of the IEEE International Conference on Image Processing*, pages 2097–2100, Orlando, Florida, September 2012.

Haichao Zhang, Jianchao Yang, Yanning Zhang, Nasser M. Nasrabadi, and Thomas S. Huang. Close the loop: Joint blind image restoration and recognition with sparse representation prior. In *Proceedings of the Thirteenth IEEE International Conference on Computer Vision*, Barcelona, Spain, November 2011.

Haichao Zhang, Yanning Zhang, Haisen Li, and Thomas S. Huang. Generative Bayesian image super resolution with natural image prior. *IEEE Transactions on Image Processing*, 21(9):4054–4067, 2012.

Li Zhang and Steven M. Seitz. Estimating optimal parameters for MRF stereo from a single image pair. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):331–342, February 2007.

Ruo Zhang, Ping-Sing Tsai, James E. Cryer, and Mubarak Shah. Shape-from-shading: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):690–706, August 1999.

Bo Zhao, Wensheng Zhang, Huan Ding, and Hu Wang. Non-blind image deblurring from a single image. *Cognitive Computation*, 5(1): 3–12, 2013.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the Fifteenth IEEE International Conference on Computer Vision*, pages 1529–1537, Santiago, Chile, December 2015.

Song Chun Zhu and David Mumford. Prior learning and Gibbs reaction-diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(11):1236–1250, November 1997.

Vladimir Zlokolica, Aleksandra Pižurica, and Wilfried Philips. Noise estimation for video processing based on spatio-temporal gradients. *IEEE Signal Processing Letters*, 13(6):337–340, June 2006.

Siavash Zokai and George Wolberg. Image registration using log-polar mappings for recovery of large-scale similarity and projective transformations. *IEEE Transactions on Image Processing*, 14(10), October 2005.

Maria Zontak and Michal Irani. Internal statistics of a single natural image. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 977–984, Colorado Springs, Colorado, June 2011.

Daniel Zoran and Yair Weiss. Scale invariance and noise in natural images. In *Proceedings of the Twelfth IEEE International Conference on Computer Vision*, pages 2209–2216, Kyoto, Japan, October 2009.

Daniel Zoran and Yair Weiss. From learning models of natural image patches to whole image restoration. In *Proceedings of the Thirteenth IEEE International Conference on Computer Vision*, 2011.

CURRICULUM VITÆ

UWE SCHMIDT

Date of birth: February 27th, 1983
Place of birth: Hanau, Germany

Education	2010 – 2015	*Technische Universität Darmstadt, Germany* Ph.D. student in Computer Science
	2006 – 2010	*Technische Universität Darmstadt, Germany* M.Sc. in Computer Science
	2006 – 2007	*University of British Columbia, Vancouver, Canada* Visiting graduate student in Computer Science
	2002 – 2006	*Technische Universität Darmstadt, Germany* B.Sc. in Computer Science
	1993 – 2002	*Franziskanergymnasium Kreuzburg, Großkrotzenburg, Germany*

Positions	Since 2015	*MPI of Molecular Cell Biology and Genetics, Dresden, Germany* Group of Gene Myers Research assistant
	2010 – 2015	*Technische Universität Darmstadt, Germany* Visual Inference group of Prof. Stefan Roth Research and teaching assistant
	2012	*Microsoft Research, Cambridge, United Kingdom* Intern at the Machine Learning and Perception group
	2006	*University of British Columbia, Vancouver, Canada* Student teaching assistant
	2004 – 2006	*Technische Universität Darmstadt, Germany* Intermittent student teaching/research assistant

PUBLICATIONS

UWE SCHMIDT, JEREMY JANCSARY, SEBASTIAN NOWOZIN, STEFAN ROTH, AND CARSTEN ROTHER
Cascades of Regression Tree Fields for Image Restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, volume 38, issue 4, pages 677-689, April 2016.

UWE SCHMIDT AND STEFAN ROTH
Shrinkage Fields for Effective Image Restoration. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Columbus, Ohio, June 2014.

UWE SCHMIDT, CARSTEN ROTHER, SEBASTIAN NOWOZIN, JEREMY JANCSARY, AND STEFAN ROTH
Discriminative Non-blind Deblurring. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Portland, Oregon, June 2013.

THORSTEN FRANZEL, UWE SCHMIDT, AND STEFAN ROTH
Object Detection in Multi-View X-Ray Images. In *Joint Pattern Recognition Symposium (34th DAGM, 36th OAGM)*, Graz, Austria, August 2012.

UWE SCHMIDT AND STEFAN ROTH
Learning Rotation-Aware Features: From Invariant Priors to Equivariant Descriptors. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Providence, Rhode Island, June 2012.

UWE SCHMIDT, KEVIN SCHELTEN, AND STEFAN ROTH
Bayesian Deblurring with Integrated Noise Estimation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Colorado Springs, Colorado, June 2011.

UWE SCHMIDT, QI GAO, AND STEFAN ROTH
A Generative Perspective on MRFs in Low-Level Vision. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, California, June 2010.

JUSSI KANGASHARJU, UWE SCHMIDT, DIRK BRADLER, AND JULIAN SCHRÖDER-BERNHARDI.
ChunkSim: Simulating Peer-to-Peer Content Distribution. In *Proceedings of the Spring Simulation Multiconference (SpringSim)*, Norfolk, Virginia, 2007.