A Bayesian Approach to Variational Methods

René Ranftl

A Bayesian Approach to Variational Methods

Master's Thesis

at

Graz University of Technology

submitted by

René Ranftl

Institute for Computer Graphics and Vision (ICG),
Graz University of Technology
A-8010 Graz, Austria

15th October 2010

Advisor: Dipl.-Ing. Dr. Thomas Pock

Bayes'sche Methoden für Variationsmodelle

Diplomarbeit

an der

Technischen Universität Graz

vorgelegt von

René Ranftl

Institut für Maschinelles Sehen und Darstellen (ICG),
Technische Universität Graz
A-8010 Graz

15. Oktober 2010

Diese Arbeit ist in englischer Sprache verfasst.

Begutachter: Dipl.-Ing. Dr. Thomas Pock

Abstract

Variational models are among the most successful methods for low-level Computer Vision tasks today. While such models can be derived and formulated in a completely deterministic setting, they nonetheless have a deep connection to the probabilistic framework of Bayesian inference. This thesis highlights this connection and the advantages that a probabilistic approach to variational methods can have.

A fundamental question in variational models is the formulation of an appropriate image model. A especially popular image model is given by the Total Variation prior due to its edge preserving properties. It will be shown that the usually employed energy minimization approach is not able to fully exploit the properties of the underlying models if such an image prior is used. An alternative approach that is based on Bayesian estimation is introduced and the connections to energy minimization are highlighted.

The proposed estimator is defined by a very high-dimensional integral that can not be solved with deterministic numerical integration algorithms. To tackle this problem, the framework of Markov Chain Monte Carlo (MCMC) integration is introduced and refined into an algorithm that is specifically tailored to the needs of image processing. To speed up the computations, a parallelization scheme and an implementation on graphics processing hardware is proposed.

The advantages of the proposed algorithm over the energy minimization approach are shown on convex image reconstruction models. For non-convex models the MCMC approach allows for global optimization. Our experiments on different models for motion estimation and stereo reconstruction show that such a global optimization approach is not only feasible but also provides superior results.

Keywords: Variational Methods, Total Variation, Estimation Theory, MCMC, Optical Flow, Stereo, Denoising, GPGPU

Kurzfassung

Variationsmethoden gehören zu den populärsten Verfahren zur Lösung einer Reihe von low-level Problemen im Bereich Computer Vision. Solche Modelle werden üblicherweise vollständig deterministisch formuliert. Bei genauer Betrachtung stellt sich heraus, dass Variationsmethoden eine Verbindung zur probabilistischen Methode der bayes’schen Modellbildung und Deduktion haben. Diese Masterarbeit zeigt diese Verbindung und Vorteile einer probabilistische Herangehensweise auf.

Ein fundamendaler Schritt in der Anwendung von Variationsmethoden, ist die Definition eines passenden Bildmodells. Total Variation ist, aufgrund der kantenerhaltenden Eigenschaften, ein weit verbreitetes Bildmodell. Es wird gezeigt, dass ein Energieminimierungsansatz die deskriptiven Eigenschaften von Modellen, die auf Total Variation basieren, nicht optimal nutzen kann. Ein alternativer bayes’scher Schätzer, der die Eigenschaften des Modells besser nutzt, und dessen Verbindung zu Energieminimierung, wird vorgestellt.

Der Schätzer ist über ein sehr hochdimensionales Integral definiert, welches mit deterministischen, numerischen Integrationsverfahren nicht gelöst werden kann. Ein probabilistischer Ansatz zur hochdimensionalen, numerischen Integration, bekannt als “Markov Chain Monte Carlo”-Integration (MCMC) wird vorgestellt. MCMC-Algorithmen sind typischerweise sehr aufwändig zu berechnen. Zur Beschleunigung der Algorithmen wird eine GPU-basierte, parallele Implementierung vorgestellt.

Die Vorteile der vorgestellten Methode werden anhand konvexer Modelle zum Entrauschen von Bildern aufgezeigt. Für nichtkonvexe Modelle können MCMC-Methoden zur globalen Optimierung benutzt werden. Anhand von Modellen zur Stereorekonstruktion und der Schätzung von Optical Flow wird gezeigt, dass globale Optimierung den üblichen Ansätzen qualitativ überlegen ist.

Deutsche Fassung:
Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008
Genehmigung des Senates am 1.12.2008

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

Graz, am …………………………… ………………………………………………..
(Unterschrift)

Englische Fassung:

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

…………………………… ………………………………………………..
date (signature)

Contents

List of Figures

Acknowledgments

First and foremost I would like to thank my supervisor, Dipl.-Ing. Dr. Thomas Pock, for countless personal discussions and his patient guidance that made this thesis possible. Special thanks go to Manuel Werlberger for technical support and Matthias Freiberger for spontaneous proofreading. I am particularly grateful for my parents for their love and support during my years of study.

René Ranftl
Graz, Austria, October 2010

Chapter 1

Introduction

The optimization of variational models is an important subdiscipline of Computer Vision. Many low-level vision tasks, such as reconstruction, deconvolution, segmentation and optical flow, all of them inherently inverse and ill-posed problems, can be formulated as energy minimization problems. The design of such algorithms can be split in two main phases: First, an energy, based on empirical observations and the task at hand, is formulated. Second, an appropriate optimization procedure is employed to find a solution to the model. The first part is the most crucial part of the design procedure. The model has to be powerful enough to capture all relevant properties of the problem at hand. However one cannot define arbitrarily complicated models. Current mathematical theory of optimization typically requires a certain form for the energy to have a solution that can be found in reasonable time (or to even have a solution at all).

Generally, a variational problem is given by an energy

$$E(u) = \mathcal{D}(u) + \mathcal{R}(u)$$

where one seeks for the image u that minimizes this energy. The energy consists of a data term $\mathcal{D}(u)$ and a regularization term $\mathcal{R}(u)$. The data term models the actual data at hand and its form is therefore mainly driven by the actual application of the model. The regularization term is chosen according to some prior knowledge about properties of desirable solutions. This knowledge is typically obtained purely from empirical observations or logical assumptions. Provided that the data term is able to model the input data sufficiently enough, the quality of the results is mainly driven by the regularization term. In the context of Computer Vision and image processing, our prior knowledge consists of some assumptions of how a natural image looks. A simplistic assumption that is often made is that natural images consist of regions of constant color that are separated by sharp jumps, called edges. Such an assumption is suitable to be used in mathematical framework as there are convenient tools to mathematically describe constant regions as well as sharp jumps.

An interesting property of such fully deterministic energy minimization problems is that they allow an alternative viewpoint. Both the data term and the regularization term can be viewed in the context of probability density functions. In that sense, the data term is attached to a probability of a given observation to match the data, and the regularization term is attached to a prior probability (i.e. the probability that an image matches the prior knowledge). Putting both probability density functions together via the well-known Bayes theorem, one gets a posterior probability density function that assigns a probability to every possible image in the context of the model:

$$p(u) = \frac{1}{Z} \exp\left\{-\frac{E(u)}{T}\right\} \tag{1.1}$$

The image minimizing the initial variational problem is then equivalent to the image maximizing the posterior probability (1.1). The implications of such a viewpoint may seem minor at first, it is however a well-known fact in the Bayesian practitioners community that the maximizer of a probability function is not the most typical candidate of the distribution under all circumstances. Moreover, this image often does not exhibit the properties that were initially modeled, rendering the initial design phase at least questionable.

This discrepancy between model and solution forms the basis of this thesis. We will show that in the variational models that are employed in Computer Vision, the image that minimizes the model is indeed not necessarily the "best" image. This opens up the possibility to enhance existing models by changing the procedure to obtain a solution from the model. We will therefore present an alternative solution strategy that is based on the sampling of the probability density function (i.e. candidate images are generated according to the law defined by the posterior distribution). Such a sampling procedure enables many opportunities: First, it is possible to compute summary statistics of the posterior distribution. We will show that a specific summary statistic, namely the expected value, offers a better solution than the image that maximizes the posterior probability. Second, also the initial optimization problem can profit from such a sampling based procedure. We will introduce a global optimization algorithm that relaxes some of the constraints that have to be typically imposed on the design of a model. The presented sampling algorithms fall under a class of algorithms that is called Markov Chain Monte Carlo algorithms, or short MCMC. Such algorithms cleverly construct a Markov Chain to generate samples from an arbitrary probability density function. MCMC algorithms, however, are relatively slow when compared to optimization algorithms. We therefore propose an implementation on graphics processing hardware to obtain solutions in reasonable time.

To show the viability of our approach, we will consider two applications: denoising of images and estimation of motion. For both applications, a variety of data terms and regularizers exist, and we will more closely examine the influence of those terms and provide a comparison to the results of the usually employed energy minimization approaches.

1.1 Related Work

In the context of image processing, energy minimization methods have gained a lot of attention since the publication of the ROF model by Rudin, Osher and Fatemi (see [Rudin et al., 1992]). The model was developed for the denoising of images that were degenerated by additive white Gaussian noise and reads

$$u^* = \min_u \int_\Omega (u-f)^2 dx + \lambda \int_\Omega |\nabla u| dx$$

where Ω is the image domain, f is the image to be denoised and u^* is the minimizer of the model. The model is able to remove noise while steep edges are preserved. This property can be attributed to the regularizer $\int_\Omega |\nabla u| dx$ that is used in the model. The expression is known as the total variation seminorm and has extensively been used in numerous other image processing applications, like structure-texture decomposition [Aujol et al., 2006], blind deconvolution [Chan and Wong, 2002] and optical flow estimation [Zach et al., 2007]. While the total variation is only defined for gray-valued images, extensions to account for vector valued images can be found in [Blomgren and Chan, 1998] and [Aujol and Kang, 2006].

The total variation regularizer is known to produce block-like artifacts (known as staircases) when used in the context of energy minimization (we will later see that this is not the case in the presented approach). Explanations for this behavior can be found in [Caselles et al., 2008]. Much of the ongoing research was devoted to the reduction of the staircasing artifacts (see for example [Savage and Chen, 2006], [Chan et al., 2007] and [Chan et al., 2005]). All of those methods change the regularizer in some way or another and stay within the framework of energy minimization. Nikolova [2007] shows why such ad-hoc modifications are not optimal. The main problem lies not within the regularizer or the model itself but instead in the inference procedure, i.e. the image minimizing the variational energy does not follow the initial properties that where explicitly modeled. No regularizer that is based on spatial derivatives and has properties that are desirable in image processing applications (i.e. allows sharp jumps in the solution) is able to exploit its full potential when used in the framework of energy minimization.

A probabilistic approach to total variation denoising was first published by Louchet [2008]. In this thesis, a rigorous mathematical justification for the inference of the expected value of the ROF model is developed.

A very popular method that competes with Total variation based models are Markov Random Fields (MRFs). Unlike variational models, the MRF based models are defined in a discrete setting. Spatial relationships between cliques of pixels (i.e. neighboring pixels) are either modeled by hand or learned by a machine learning algorithm. Such models are inherently probabilistic as the spatial relationships are modeled as probability density functions. MRFs have a long history in Computer Vision, dating back to the first applications of the famous Ising model for the denoising of binary images. Notable algorithms that fall into the MRF category are Fields of Experts [Roth and Black, 2009] and Gaussian Conditional Random Fields [Tappen et al., 2007]. Fields of Experts was recently extended to the inference of the expected value with convincing results [Schmidt et al., 2010].

When viewing total variation based models in a discrete and probabilistic setting, a close relationship between the total variation norm and MRF based models becomes apparent. After discretization the total variation regularizer also models spatial relationships between directly neighboring pixels using a probability density function. In some way, the discretization of variational models can therefore also be seen as a special case of MRF based models.

1.2 Organization of this Thesis

This thesis is organized as follows:

In chapter 2, a simple variational denoising model, based on probabilistic arguments, is introduced. Standard techniques for the local optimization of this model and the importance of correct modeling of prior knowledge are demonstrated using two well-known instances of the presented model. Based on the previous probabilistic derivation of the model, we show that the deterministic optimization of a variational model can also be considered as a probabilistic estimation problem, known as Maximum-A-Posteriori estimation (MAP).

Chapter 3 shows problems and shortcomings of the MAP approach. Using our previous stochastic derivation, we propose an alternative inference procedure that is known as least-squares estimation in literature. This inference procedure, however, is far more difficult to implement than the standard MAP approach because it relies on the integration of a very high-dimensional integral.

To solve said integral, we introduce a stochastic technique for approximate high-dimensional integration in chapter 4. This approach is based on the construction of Markov Chains to sample probability density functions and generally known as Markov Chain Monte Carlo (MCMC). We provide an introduction to general state-space Markov Chains and present some of the most important algorithms of the MCMC family.

Chapter 5 is concerned with the development of efficient algorithms for our specific applications. Moreover, we introduce a global optimization procedure that is also based on the previously introduced samplers.

Details for the implementation of the algorithms on graphics processing hardware are provided in chapter 6. Starting with an overview of general purpose computing on graphic-processing hardware, a simple design along with the most outstanding obstacles for a parallel implementation are discussed.

Chapter 7 shows applications and experimental evaluations of our approach. Different denoising models as well as the estimation of optical flow and stereo stereo reconstruction are covered.

Finally, chapter 8 summarizes our findings and gives a conclusion and a short outlook on unsolved problems.

Chapter 2

Optimization in Computer Vision

Optimization generally refers to a procedure of finding the best solution to a problem among its set of possible solutions. More precisely, optimization in mathematics refers to algorithms that find the extrema of a function $f(x)$ subject to some constraints. Many problems in Computer Vision, among others Denoising, Blind Deconvolution, Stereo Reconstruction and Optical Flow, can be formulated as energy minimization problems and are therefore subject to some optimization procedure.

The chapter at hand is organized as follows: Section 2.1 introduces a simple class of denoising model based on probabilistic arguments, that will serve as an exemplar optimization problem for the rest of the chapter. Section 2.2 shows the optimization of two particular (convex) instances of the previously defined optimization problem. Finally, Section 2.3 gives a brief summary of the concepts developed in this chapter.

2.1 A Denoising Model

Denoising is the task of reconstructing an image that was corrupted by some kind of noise. As many high-level Computer Vision algorithms are sensitive to noise, the denoising of an input image is an important preprocessing step in image processing applications. A class of denoising algorithms that is able to eliminate additive white Gaussian noise is developed in this section.

Let us assume, the image has been corrupted according to an additive degradation model:

$$f(x,y) = u(x,y) + n(x,y) \tag{2.1}$$

where $f : D \to \mathbb{R}$ is the observed degraded image intensity, $u : D \to \mathbb{R}$ is the original undegraded image intensity and D is the discrete set of pixels in the rectangular image domain, i.e.:

$$D = \{(x_i, y_j) = (ih, jh) | 1 \leq i \leq N, 1 \leq j \leq M\} \tag{2.2}$$

for some grid spacing h.

$n : D \to \mathbb{R}$ denotes the degradation noise and is assumed to consist of identical and independently distributed (i.i.d) samples from a white Gaussian distribution (i.e. $n(x,y) \sim \mathcal{N}(0, \sigma^2)$).

A denoising model now aims at recovering the original image u from a given observation f. This is clearly an inverse problem since many clean images could have led to the same noisy image.

Noise is by definition a stochastic quantity. It is therefore very natural to consider a probabilistic approach:

Let us treat the images f and u from (2.1) as realizations of two random variables U and F. Both random variables have a probability density function (pdf) $p_U(u)$ and $p_F(f)$ attached, i.e.: $p_U(u)$ and

$p_F(f)$ describe the likelihood for a given realization of u and f respectively to occur. For notational simplicity, we will further omit the subscripts for pdfs and write $p(u)$ and $p(f)$ instead.

Using conditional probabilities and Bayes' Theorem, the likelihood that an image u is the undegraded image in the context of the model, given an observation f can be expressed as

$$p(u|f) = \frac{p(f|u)p(u)}{p(f)} \tag{2.3}$$

where $p(u|f)$ is called the posterior probability density function and $p(f|u)$ denotes the probability that f was generated from u. We will further call this quantity the data term because it essentially encodes the data generation model. $p(u)$ is called the prior pdf and encodes prior knowledge about desirable solutions. Note that $p(u)$ can in principle be freely chosen. Typical approaches for the derivation of a prior pdf in image processing are based on distributions of spatial derivatives or responses to local filters. It will become clear that certain choices for the prior term allow to capture properties of images better than another.

Let us now derive explicit representations for the quantities in (2.3) for an image denoising model:

Given the linear degradation model in (2.1) and under the assumption that the noise is white, Gaussian and i.i.d., it is easy to derive an expression for $p(f|u)$:

$$p(f|u) = \prod_{x,y \in D} \frac{1}{\sqrt{2\pi\sigma^2}} \exp - \frac{(f(x,y) - u(x,y))^2}{2\sigma^2} \tag{2.4}$$

where σ^2 denotes the variance of the noise.

While it was already noted that the prior $p(u)$ can in principle be arbitrarily chosen, it turns out that it is crucial for the quality of reconstruction. Huang and Mumford studied the statistical relationships of pixels in natural images in [Huang and Mumford, 1999]. They tried to fit probability density functions to capture the inter-pixel relationship of directly neighboring pixels.

Let us denote the discrete approximation of the gradient operator, applied to the image u, by ∇u. It turns out that a generalized Laplacian distribution

$$p(u) = \frac{1}{Z} \prod_{x,y \in D} \exp - \frac{|(\nabla u)(x,y)|^p}{\beta} \tag{2.5}$$

with $p = 0.55$ results in a good fit to the spatial relationships of natural images. (2.5) is normalized by some constant Z to ensure that $\int p(u)du$ integrates to one and has a parameter β that controls the spread of the probability density function. Note that setting $p = 1$ yields a standard Laplacian distribution while setting $p = 2$ results in a Gaussian distribution. Figure 2.1 shows a logarithmic plot of the generalized Laplacian distribution for different values of p for the one-dimensional case.

Putting everything together, one arrives at an explicit representation for the posterior probability density function:

$$p(u|f) = \frac{1}{Z(f)} \prod_{x,y \in D} \exp - \frac{(f(x,y) - u(x,y))^2}{2\sigma^2} \prod_{x,y \in D} \exp - \frac{|(\nabla u)(x,y)|^p}{\beta} \tag{2.6}$$

where the function $Z(f)$, the partition function, gathers $p(f)$ and all normalizing constant, in such a way that $\int p(u|f)du$ again integrates to one. Note that the product over the individual pixels can be represented by a sum in the exponent:

$$p(u|f) = \frac{1}{Z(f)} \exp - \frac{\sum_{x,y \in D}(f(x,y) - u(x,y))^2}{2\sigma^2} + \frac{\sum_{x,y \in D} |(\nabla u)(x,y)|^p}{\beta} \tag{2.7}$$

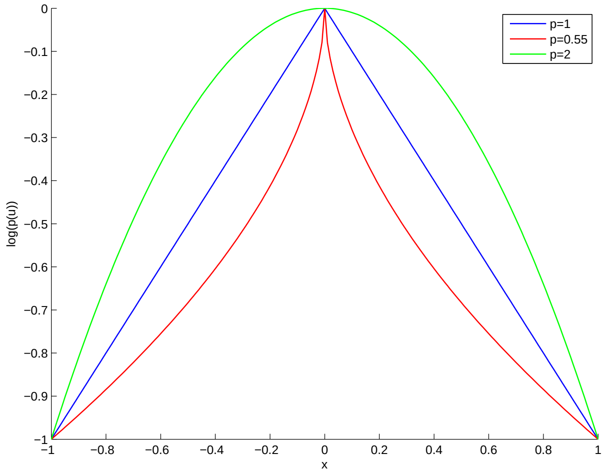


Figure 2.1: Logarithmic plot of generalized Laplacian distribution for different values of p and fixed $\beta = 1$.

(2.7) assigns a probability with respect to the model to every possible image. A natural method to find an image u that very likely belongs to the observation f, is the Maximum-A-Posteriori estimation: Find the image u^* that maximizes the posterior probability $p(u|f)$. Formally, this can be stated as

$$u^* = \arg\max_u p(u|f)$$

which results, when applied to (2.7), in

$$u^* = \arg\min_u \sum_{x,y \in D} (f(x,y) - u(x,y))^2 + \lambda \sum_{x,y \in D} |(\nabla u)(x,y)|^p \tag{2.8}$$

where the parameters σ^2 and β have been accumulated to a single parameter $\lambda = \frac{2\sigma^2}{\beta}$.

While any actual implementation of an algorithm that solves (2.8) on a digital computer has to operate in a discrete setting, it is advantageous to carry out an analysis of such functionals in continuous space. The continuous representation allows a convenient analysis of properties of the model, like existence and uniqueness of solutions. Moreover it is possible to postpone discretization to latter stages of the algorithm design, which allows greater flexibility in the choice of a particular discretization scheme.

Recall the discrete image domain given in (2.2). By letting $h \to 0$, (i.e. the spacing between two adjacent pixels becomes infinitesimally small), we obtain a continuous image domain $\Omega \subseteq \mathbb{R}$. Consequently, one has to replace sums by integrals. This leads to a continuous analog of (2.8):

$$u^* = \arg\min_u \frac{1}{2} \int_\Omega (f-u)^2 dx + \lambda \int_\Omega |\nabla u|^p dx \tag{2.9}$$

The solution of the model in (2.9) strongly depends on the actual choice of the parameter p. The optimal choice $p = 0.55$ in terms of statistical relationships between pixels in natural images results in a non-convex functional, which makes the optimization of the model very difficult. It is therefore a rather uncommon choice.

The two popular choices are $p = 2$, resulting in the Tikhonov model [Tikhonov and Arsenin, 1977], and $p = 1$, resulting in the ROF model [Rudin et al., 1992]. Both models are convex and therefore relatively simple to optimize. We will see however that the ROF model is clearly superior to the Tikhonov model in the context of image processing in the following section.

2.2 Convex Optimization

An optimization problem in its most general mathematical form is given by:

$$\begin{array}{lll} \text{Minimize} & f(x), & x \in X \subseteq \mathbb{R}^d \\ \text{Subject to} & g_i(x) \leq 0, & i = 1, \ldots, n \end{array} \tag{2.10}$$

where $f : X \to \mathbb{R}$ is called the objective function and the functions $g_i : X \to \mathbb{R}$ are called the constraint functions.

Any x^* that satisfies $f(x^*) \leq f(x),\ \forall x \in X$ and $g_i(x^*) \leq 0,\ i = 1, \ldots, n$ is called globally optimal and is therefore a solution of (2.10).

Numerous algorithms for optimization are not able to reliably determine the global optimum of a problem. Such algorithms are called local optimization algorithms. Given that there are local optima, i.e. optima that satisfy $f(x^*) \leq f(x)$ only in some neighborhood of x^*, such algorithms can get stuck in local optima depending on the initialization of the algorithm. To make things worse, one is not even able to tell whether the optimum that was found is global or not. Local optimization algorithms are, however, very fast compared to algorithms that guarantee to find the global optimum, which is the key to their large popularity. Moreover, if all local optima are global optima, one can be confident that any local optimization algorithm is able to find a global optimum. The most general class of problems where all local optima are global as well are convex problems.

Let us first formalize the notion of convexity:

Definition 1. *([Boyd and Vandenberghe, 2004]) A function $f : X^d \to \mathbb{R}$ is called convex, if it satisfies for all $x, y \in X^d$ and for any $\alpha, \beta \in (0, 1)$ with $\alpha + \beta = 1$:*

$$f(\alpha x + \beta y) \leq \alpha f(x) + \beta f(y)$$

and X^d is a convex set. We call the function concave, if $-f(x)$ is convex.

An optimization problem where the objective function $f(x)$ as well as all constraint functions $g_i(x)$ are convex is called a convex optimization problem.

The optimization of convex functionals is a sub-discipline of optimization that is very well developed. There are numerous efficient algorithms to solve such problems and a multitude of tools to approximate non-convex problems by convex ones (see for example [Boyd and Vandenberghe, 2004] and [Nesterov, 2004]).

Let us now check for which values of p the functional defined in (2.9) is convex. The energy $E(u; \lambda, p)$ is given by:

$$E(u; \lambda, p) = \frac{1}{2} \int_\Omega (f - u)^2 dx + \lambda \int_\Omega |\nabla u|^p dx \tag{2.11}$$

Sums of convex functionals are again convex. It therefore suffices to show convexity for the data term and the prior individually.

It is easy to verify that the data term $\int_\Omega (f - u)^2 dx$ is indeed convex with respect to u.

The prior resembles the p-th power of a p-norm, i.e. $\int_\Omega |\nabla u|^p dx = \|\nabla u\|_p^p$, where the p-norm is defined as:

$$\|f(x)\|_p = \left(\int_\Omega |f(x)|^p dx \right)^{\frac{1}{p}} \tag{2.12}$$

To see for which values of p (2.12) is convex (and therefore a true norm), consider the definition of convexity (Definition 1). Using two test functions $f, g : \mathbb{R} \to \mathbb{R}$, where $g(x) = 0$ for all $x \in \mathbb{R}$, we can derive the relation

$$\|\alpha f + \beta g\|_p^p = \alpha^p \|f\|_p^p \leq \alpha \|f\|_p^p = \alpha \|f\|_p^p + \beta \|g\|_p^p$$

which is only true if $\alpha^p \leq \alpha$. This is clearly only the case for $p \geq 1$. Choosing p smaller than one therefore always results in a non-convex functional.

Via Minkowski's inequality, it can easily be seen that for the remaining case $p \geq 1$ the functional is always convex:

$$\|\alpha f + \beta g\|_p^p \leq \|\alpha f\|_p^p + \|\beta f\|_p^p = \alpha^p \|f\|_p^p + \beta^p \|g\|_p^p \leq \alpha \|f\|_p^p + \beta \|g\|_p^p$$

This shows the functional (2.11) is only convex for $p \geq 1$ (and positive λ).

The two popular choices $p = 1$ and $p = 2$ are therefore both convex and can readily be optimized using local optimization methods.

2.2.1 Tikhonov Regularization

The first important special case that is examined is (2.9) with $p = 2$, resulting in the well-known Tikhonov model:

$$E(u; \lambda) = \frac{1}{2} \int_\Omega (f - u)^2 dx + \frac{\lambda}{2} \int_\Omega |\nabla u|^2 dx \tag{2.13}$$

In order to minimize the energy given in (2.13), we need some means to describe the minimum of the functional. Similar to the standard approach of vector-analysis, we are looking for points where the functional is stationary. The Euler-Lagrange equations provide a convenient tool to describe such stationary points in functional analysis:

Given a functional of the form

$$E(u) = \int_\Omega F(x, u, \nabla u) dx$$

the Euler-Lagrange equations allow us to describe the dynamics of any such functional at its stationary points by a differential equation:

$$\frac{\partial E(u)}{\partial u} = \frac{\partial F(x, u, \nabla u)}{\partial u} - \nabla \left(\frac{\partial F(x, u, \nabla u)}{\partial (\nabla u)} \right) = 0 \tag{2.14}$$

(2.14) can be understood as the functional analog of gradients. It can therefore be used to move "downhill" in a functional, within the framework of methods of steepest descent, i.e

$$u_{t+1} = u_t - \gamma \frac{\partial E(u)}{\partial u}\Big|_{u=u_t} \tag{2.15}$$

implies that $u_{t+1} \leq u_t$ provided that $0 < \gamma < \frac{2}{K}$, where K denotes the Lipschitz constant of $\frac{\partial E(u)}{\partial u}$:

$$\left|\left| \frac{\partial E(u)}{\partial u} - \frac{\partial E(v)}{\partial v} \right|\right| \leq K \, ||u - v||$$

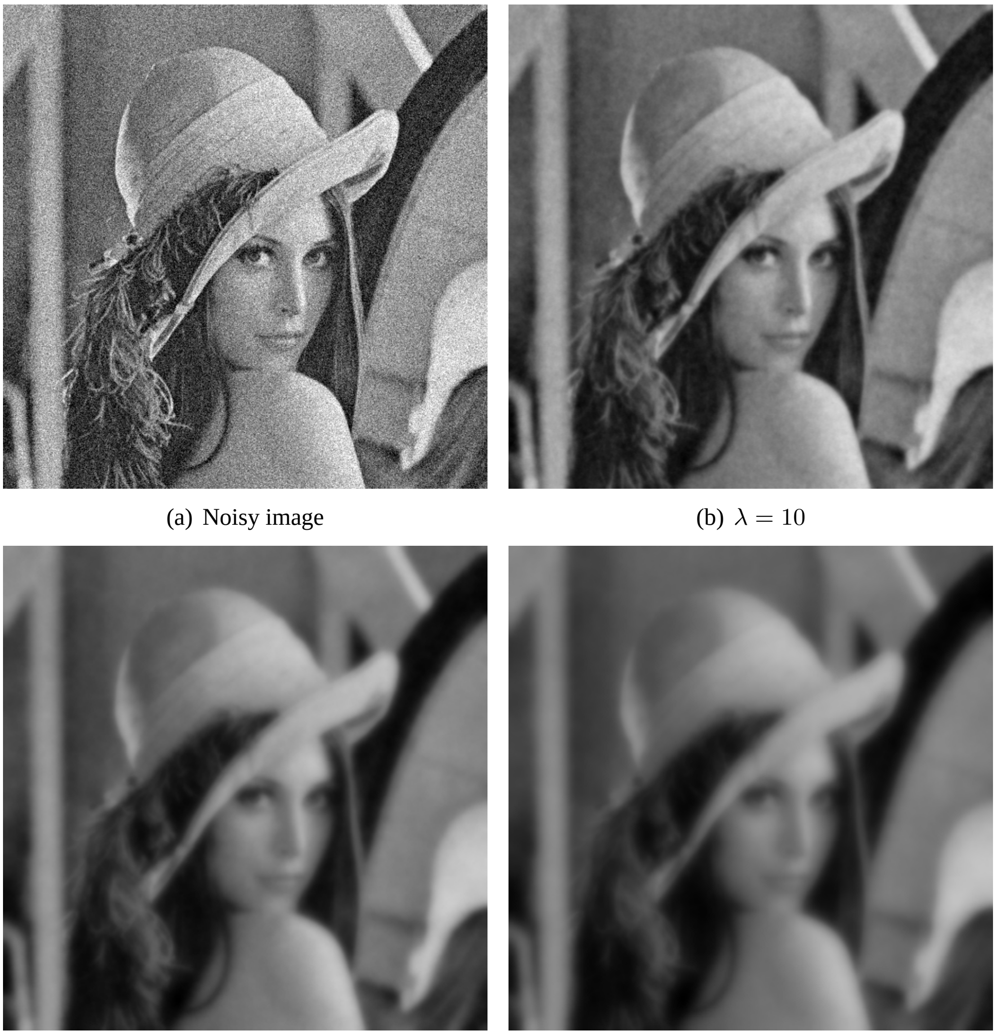
Based on this relation, it can be shown that the optimal constant step-size is given by $\gamma = \frac{1}{K}$ ([Nesterov, 2004]).

Any stationary point in a convex functional coincides with a global optimum. This guarantees that the gradient descent scheme in (2.15) yields $\lim_{t \to \infty} u_t = u^*$.

Let us now turn to the optimization of the Tikhonov model. Application of (2.14) to (2.13) yields that the minimum u^* satisfies:

$$\frac{\partial E(u)}{\partial u}\Big|_{u^*} = (u^* - f) - \lambda \text{div}(\nabla u^*) = 0 \tag{2.16}$$

Using (2.16) together with (2.15) provides a simple algorithm for the optimization of the Tikhonov model.



(a) Noisy image

(b) $\lambda = 10$

(c) $\lambda = 50$

(d) $\lambda = 100$

Figure 2.2: Reconstruction of a noisy image using the Tikhonov model for different values of λ. Noise disappears for larger λ, as do small-scale structures and edges.

Figure 2.2 shows results of the Tikhonov model applied to a noisy image for different values of λ. Higher values of λ give stronger regularization of the noisy image. It is apparent, however, that for high λ, image details become blurred too. To explain this effect, note that an explicit solution to (2.16) can be computed:

$$u - \lambda(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) = f$$

Using the Fourier transform, we get

$$\hat{u} + \lambda(4\pi^2\eta_x^2\hat{u} + 4\pi^2\eta_y^2\hat{u}) = \hat{f}$$

where $\hat{g}(\eta_x, \eta_y)$ denotes the Fourier transform of function $g(x, y)$, with spatial frequencies η_x and η_y. This relates the image u to the input f as:

$$\hat{u} = \frac{1}{1 + 4\pi^2\lambda(\eta_x^2 + \eta_y^2)}\hat{f} = \hat{K}(\lambda)\hat{f} \tag{2.17}$$

in the Fourier domain.

A fundamental property of the Fourier transform states that multiplication in the Fourier domain is equivalent to convolution in the spatial domain. This allows us to finally express the solution as

$$u = K(\lambda) * f \tag{2.18}$$

where $K(\lambda)$ is obtained from the inverse Fourier transform of $\hat{K}(\lambda)$. (2.18) shows that the solution to the Tikhonov model can be obtained by linear filtering of the input image f by a convolution kernel $K(\lambda)$. (2.17) gives some insight into the nature of this operation. The Fourier transform of the convolution kernel $K(\lambda)$ is

$$\hat{K}(\lambda) = \frac{1}{1 + 4\pi^2\lambda(\eta_x^2 + \eta_y^2)}$$

which has the typical form of a low-pass filter (see Figure 2.3), explaining the smoothing properties. Note that the kernel does not depend on the image f, which further explains the isotropic behavior of the smoothing.

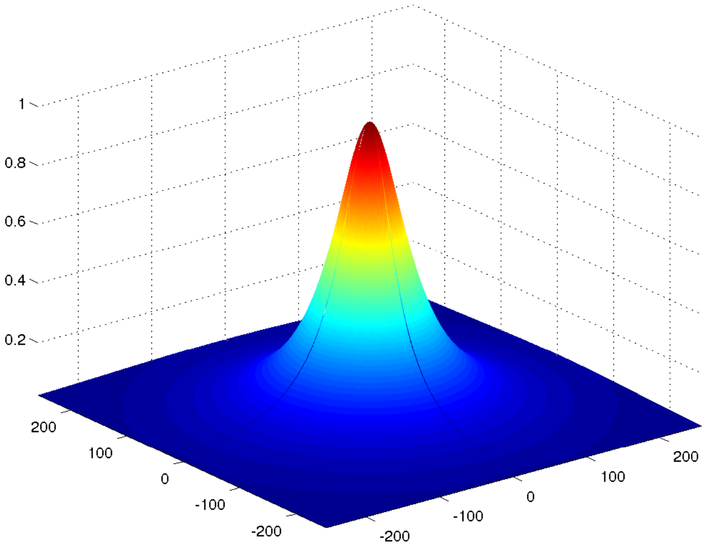


Figure 2.3: Minimization of the Tikhonov model is equivalent to low-pass filtering. The image shows the Frequency response of the corresponding kernel.

Closed form solutions to the models defined by (2.9) are not generally available for $p \neq 2$ (especially not for the very important case $p = 1$), the introduction of an optimization procedure is therefore well justified.

The gradient descent scheme that has been described so far, suffers from some severe problems. First, the algorithm is sensitive to the choice of the step-size γ. For some functionals, it may be hard to determine the Lipschitz constant or worse it may not even exist. While the optimization procedure still works in principle, the user is bound to choosing a step-size by guessing. While a step-size that is too large may yield unsatisfactory results because the algorithm oscillates around the optimum, choosing γ too small results in poor convergence time.

Moreover, the algorithm always moves into the direction of the steepest gradient with some fixed step-size. This may result in a “zig-zag” movement for some functionals, which again results in poor convergence time.

Below, we will therefore introduce an algorithm that is based on duality arguments and addresses these problems. The algorithm will be applied to the ROF model, which is far better suited in the context of image processing than the Tikhonov model.

2.2.2 The ROF Model

Consider the ROF energy:

$$E(u;\lambda) = \frac{1}{2}\int_\Omega (f-u)^2 dx + \lambda \int_\Omega |\nabla u| dx \tag{2.19}$$

with the corresponding Euler-Lagrange equation

$$(u^* - f) - \lambda \nabla (\frac{\nabla u^*}{|\nabla u^*|}) = 0 \tag{2.20}$$

It is immediately obvious that (2.20) is not defined where $|\nabla u| = 0$. Moreover, we note that (2.19) is only well defined if the gradient ∇u exists and $|\nabla u|$ is integrable in the domain Ω. More formally we have that the energy is well defined for functions $u \in W^{1,1}(\Omega)$, where $W^{1,1}(\Omega)$ denotes the space of absolutely continuous functions. As this is a rather strict regularity condition, this observation gives rise to the question which classes of functions actually can be inputs to the functional (2.19) and consequently can be solutions to the optimization problem.

Clearly, the prior dominates the space of functions for which (2.20) is well defined. The prior $\int_\Omega |\nabla u| dx$ is known as the Total Variation semi-norm $TV(u)$. Close inspection naturally yields two regularity conditions for u:

1. u has to be continuously differentiable
2. The integral over ∇u has to be bounded, in other words $TV(u) < \infty$

The first condition is a rather strict regularity condition, effectively excluding images with sharp jumps from the functional space. Fortunately, it is possible to eliminate this condition by using an alternative definition of the Total Variation semi-norm that is based on duality properties of convex functionals:

The Fenchel-Legendre Transform of the 1-norm $f(z) = \|z\|_1$ is given by

$$f^*(p) = \sup\left\{p^T.z - \|z\|_1\right\} = \begin{cases} 0 & \text{if } \|p\|_\infty \leq 1 \\ \infty & \text{else} \end{cases} \tag{2.21}$$

For convex and lower semi-continuous functionals, the Legendre-Fenchel transform is its own inverse [Boyd and Vandenberghe, 2004], thus successive application yields an alternative, so-called dual, definition of the Total Variation semi-norm:

$$TV(u) = TV^{**}(p) = \sup\left\{\int_\Omega p^T.\nabla u dx,\ \|p\|_\infty \leq 1\right\} \tag{2.22}$$

with $p : \Omega \to \mathbb{R}^2$.

By restricting p to be continuously differentiable, integration by parts yields

$$\int_\Omega p^T \nabla u dx = \int_\Gamma u p d\Gamma - \int_\Omega u \text{div} p\, dx \tag{2.23}$$

where Γ denotes the boundary of Ω and the operator div denotes the divergence operator, i.e. for a function $g(x) = (g_1(x), g_2(x))^T$ we have

$$\text{div} g = \frac{\partial g_1}{\partial x} + \frac{\partial g_2}{\partial y}$$

By further restricting p to have compact support in Ω, the integration along the boundary of Ω becomes zero, which finally yields:

$$TV(u) = \sup \left\{ -\int_\Omega u \mathrm{div} p \, dx, \; \|p\|_\infty \leq 1 \right\} \tag{2.24}$$

This definition removes the need for u to be continuously differentiable, leveraging the set of possible solutions to a set that incorporates sharp discontinuities, which is clearly advantageous for the denoising of images (and Computer Vision applications in general). We call functions with $TV(u) < \infty$ functions of bounded variation. More formally, the space of functions with bounded variation is given by

$$BV = \left\{ u \in L^1(\Omega) : TV(u) < \infty \right\} \tag{2.25}$$

From (2.22), it is easy to see the nature of the function p. To attain the supremum, one has $p(x) = \frac{(\nabla u)(x)}{|(\nabla u)(x)|}$ if $(\nabla u)(x) \neq 0$. For the case $\nabla u = 0$, $p(x)$ can be arbitrarily defined on the unit ball B_0^1, because the integrand in (2.22) is zero, independent of $p(x)$. Figure 2.4 shows a graphical depiction of the relation between the primal u and the dual p in the ROF model for a single pixel.

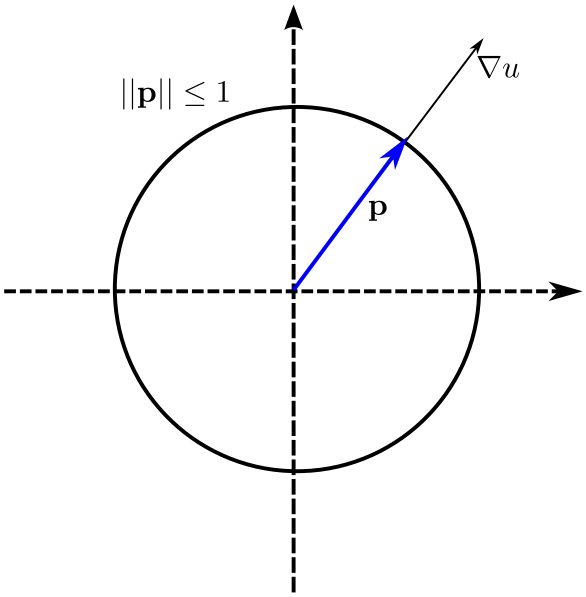


Figure 2.4: Relationship of the dual variable p to the primal ∇u for a single pixel if $\nabla u \neq 0$. p is given by normalizing ∇u to unit length.

We are now able to state the ROF functional in its primal-dual formulation

$$u^* = \min_u \max_{\|p\|_\infty \leq 1} \left\{ \frac{1}{2} \int_\Omega (u-f)^2 dx + \lambda \int_\Omega \nabla u \cdot p dx \right\} \tag{2.26}$$

Note that the relaxation of regularity constraints turns the previously unconstrained minimization problem (2.19) into a constrained saddle point problem. An important theorem in convex analysis states that the minimum and maximum operations in saddle point problems can be interchanged provided that one of the optimization operations is constrained to a compact set. As this is clearly the case in (2.26), the optimality condition with respect to u is given by

$$u = f + \lambda \mathrm{div} p \tag{2.27}$$

By substituting (2.27) into (2.26) and flipping the sign of the functional to turn the maximization problem into a minimization, one arrives at the dual formulation of the ROF model:

$$p^* = \min_{\|p\|_\infty \leq 1} \left\{ \frac{\lambda}{2} \int_\Omega (\mathrm{div} p)^2 dx + \int_\Omega f \mathrm{div} p \, dx \right\} \tag{2.28}$$

where the minimizer u^* can be recovered from p^* using (2.27).

Keeping all three proposed formulations of the ROF model in mind, we now turn to the optimization of the model. An implementation of any optimization procedure on a digital computer has to be carried out in a discrete setting due to finite memory. A crucial first step in the optimization procedure is therefore to provide a stable discretization scheme for the continuous model. Let us start with the discretization of the image domain Ω.

It is again assumed that the image is defined on the rectangular grid of size MxN with equally spaced pixel, as shown in (2.2).

The discrete analogs to u and p respectively can be expressed by the vectors $u^h \in \mathbb{R}^{MN}$ and $p^h \in \mathbb{R}^{2MN}$, which result from lexicographical stacking of the rows of the images. The most important part of the discretization scheme is the definition of appropriate discrete differentiation operators. Let us stick to a very simple definition based on finite forward and backward differences ([Chambolle, 2004]):

Finite first-order forward differences in first x and y direction are defined as

$$\begin{aligned} \partial_x^+ u_{i,j}^h &= \begin{cases} \frac{u_{i+1,j}^h - u_{i,j}^h}{h} & \text{if } 0 < i < M \\ 0 & \text{else} \end{cases} \\ \partial_y^+ u_{i,j}^h &= \begin{cases} \frac{u_{i,j+1}^h - u_{i,j}^h}{h} & \text{if } 0 < j < N \\ 0 & \text{else} \end{cases} \end{aligned}$$

and the discrete gradient operator ∇ can then be defined as

$$\nabla u_{i,j}^h = (\partial_x^+ u_{i,j}^h, \partial_y^+ u_{i,j}^h)^T$$

The gradient operator is a linear mapping, hence the differentiation of the whole image u^h can be expressed as a matrix-vector product:

Let A be the matrix of size MNx2MN that, applied to a vector of length MN, yields the $2MN$-vector containing the finite forward differences, then the discretized primal-dual ROF model can be expressed as:

$$E(u^h) = \left\langle Au^h, p^h \right\rangle + \frac{1}{2}\|u^h - f^h\|_2^2 \tag{2.29}$$

Using these basic definitions, it is possible to derive the discrete divergence operator:

(2.22) and (2.24) provide the identity

$$\int_\Omega p.\nabla u dx = -\int_\Omega u \text{div} p \, dx$$

which has to be reflected in the discretization scheme as well, i.e. we have to choose the matrix A^* of size 2MNxMN, representing the discrete divergence operator, such that

$$\left\langle Au^h, p^h \right\rangle = \left\langle u^h, A^* p^h \right\rangle \qquad \forall u^h, p^h$$

Generally speaking, A^* is the hermitian adjoint operator to A, which is guaranteed to exist for any linear mapping.

From the perspective of point-wise differentiation, it is easy to check that given the backward differences

$$\partial_x^- u_{i,j}^h = \begin{cases} \frac{u_{i,j}^h - u_{i-1,j}^h}{h} & \text{if } 1 < i < M \\ \frac{u_{i,j}^h}{h} & \text{if } i = 1 \\ \frac{-u_{i-1,j}^h}{h} & \text{if } i = M \end{cases} \qquad \partial_y^- u_{i,j}^h = \begin{cases} \frac{u_{i,j}^h - u_{i,j-1}^h}{h} & \text{if } 1 < j < N \\ \frac{u_{i,j}^h}{h} & \text{if } j = 1 \\ \frac{-u_{i,j-1}^h}{h} & \text{if } j = N \end{cases}$$

we can define the discrete divergence operator as

$$-\text{div}p^h_{i,j} = \partial^-_x p^1_{i,j} + \partial^-_y p^2_{i,j}$$

to fulfill the requirement of adjointness.

To allow a clean notation and without loss of generality, it will be assumed for the remainder of this section that the pixel spacing h is equal to one. We therefore write u instead of u^h.

Based on [Zhu and Chan, 2008], the primal-dual formulation allows us to employ an efficient class algorithms. The presented algorithm will later serve as gold-standard for MAP inference problems.

In its basic form, the primal-dual algorithm states that the generalized saddle point problem

$$\min_u \max_p \langle p, Au \rangle + G(u) - F^*(p)$$

can be optimized using simultaneous gradient descent/ascent in the primal and dual variables respectively, resulting in the following update scheme [Pock et al., 2009]:

$$\begin{aligned} p^{n+1} &= (I + \tau_d \partial F^*)^{-1}(p^n + \tau_d A\tilde{u}^n) && (2.30)\\ u^{n+1} &= (I + \tau_p \partial G)^{-1}(u^n - \tau_p A^* p^{n+1}) && (2.31)\\ \tilde{u}^{n+1} &= 2u^{n+1} - u^n && (2.32) \end{aligned}$$

for some appropriate step-sizes τ_d and τ_p. Note that there exist several variants of this algorithm (see [Chambolle and Pock, 2010] and [Esser, 2010]).

$(I + \tau_d \partial F^*)^{-1}(p^{n+\frac{1}{2}})$ denotes the resolvent operator with respect to a function F^* and is given by:

$$p^{n+1} = \arg\min_p \left\{ \frac{1}{2\tau_d} \|p - p^{n+\frac{1}{2}}\|_2^2 + F^*(p) \right\}$$

For the dual variable in the primal-dual ROF model (2.26) the function $F^*(p)$ captures the constraint that any feasible p has to be a member of the set $K = \{p : \|p\|_\infty \leq 1\}$. This can be achieved using an indicator function $F^*(p) = I_K(p)$, where $I_K(p) = 0$ if $p \in K$ and $I_K(p) = \infty$ otherwise.

This yields the dual update rule

$$p^{n+1} = \text{Prow}_K \left(p^n + \tau_d \lambda \nabla \tilde{u} \right)$$

where the operator $\text{Proj}_K(.)$ denotes an Euclidean projection onto the set K. This set is a relatively simple convex set (i.e. the unit ball in $\mathbb{R}^2$), therefore the projection can be achieved by a point-wise operation:

$$\text{Proj}_K(p(x)) = \frac{p(x)}{\max\{1, \|p(x)\|_2\}}$$

For the primal variable $G(u)$ is given by the data term, i.e. $G(u) = \frac{1}{2}\|u - f\|_2^2$. Application of the resolvent operator yields the update rule for the primal variable:

$$u^{n+1} = \frac{u^n + \tau_p(f + \lambda \text{div} p^{n+1})}{1 + \tau_p}$$

The last step (2.32) facilitate a simple extrapolation based on the current and previous iterates.

One problem in any optimization procedure is to decide when to stop the algorithm. A typical approach is to stop when the distance between two consecutive updates is below some threshold. In primal-dual algorithms it is convenient to measure the difference between the primal and dual energies [Zhu et al., 2008]:

The dual energy can be written as

$$E(p) = \frac{1}{2}\left(\int_\Omega f^2 dx - \int_\Omega (\lambda \text{div} p + f)^2 dx\right)$$

and the gap $E(u) - E(p)$ is then given by:

$$\text{GAP}(u,p) = \int_\Omega \lambda(|\nabla u| - u\text{div}p)dx + \frac{1}{2}\int_\Omega (\lambda \text{div}p + f - u)^2 dx$$

Note that the second term in the expression above is always zero, because u is chosen according to (2.27). From the same equation we have that

$$\begin{aligned} \frac{1}{\lambda}\|u - u^*\|_2^2 &= \int_\Omega (u - u^*)(\text{div}p - \text{div}p^*)dx \\ &= -\int_\Omega \nabla u \cdot p + \int_\Omega \nabla u \cdot p^* + \int_\Omega \nabla u^* \cdot p - \int_\Omega \nabla u^* \cdot p^* \end{aligned} \tag{2.33}$$

With the relations $|\nabla u^*| = \nabla u^* p^*$ and $|\nabla u| \geq \nabla u \cdot p$, it is easy to see that the gap is a bound for the distance to the minimizer u^*:

$$\frac{1}{\lambda}\|u - u^*\|_2^2 \leq -\int_\Omega \nabla u \cdot p + \int_\Omega |\nabla u| = \frac{1}{\lambda}G(u,p)$$

This shows that the Euclidean distance from u to the global optimum u^* is never larger than $\sqrt{\text{GAP}(u,p)}$ and justifies the use of the gap between the primal and dual energies as a stopping criterion.

Algorithm 1 summarizes the whole optimization procedure for the ROF model.

Algorithm 1 Primal-dual algorithm for the ROF model

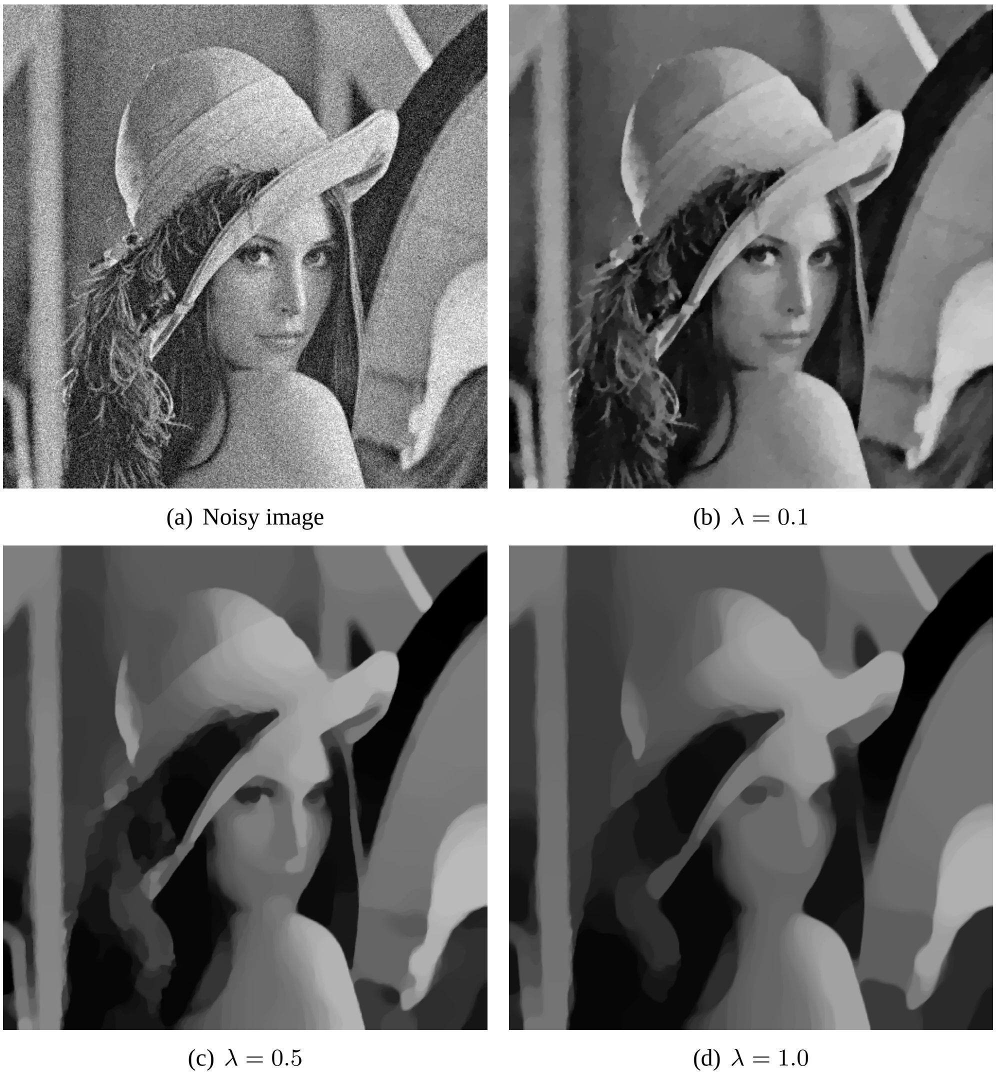
```
Set $n = 0$
Set $u^0 = f$, $\tilde{u}^0 = f$
Set $p^0_{i,j} = 0 \qquad 0 < i < M,\ 0 < j < N$
**while** GAP$(u^n, p^n) < \epsilon$ **do**
    $p^{n+1} \leftarrow \text{Proj}_K\left(p^n + \tau_d \lambda \nabla \tilde{u}\right)$
    $u^{n+1} \leftarrow \frac{u^n + \tau_p(f + \text{div}\, p^{n+1})}{1 + \tau_p}$
    $\tilde{u}^{n+1} \leftarrow 2u^{n+1} - u^n$
    $n \leftarrow n + 1$
**end while**
```

Figure 2.5 shows the results of the ROF model for different values of λ. In contrast to the Tikhonov model, edges are preserved and the overall result is visually much more appealing than the Tikhonov model.

The figure also shows an interesting effect if one increases the value of λ. As λ increases (i.e. more denoising is applied to the image), the result looks more and more like a cartoon of the original image. At closer observation, it turns out that solutions of the ROF model tend to be composed of piecewise-constant functions. This results in block-like artifacts that are, due to their shape, called staircases.

The effect is most easily observed for one-dimensional signals. Figure 2.6 shows a noisy one-dimensional signal, superimposed with the original signal (dashed line), on the left. The reconstruction is plotted on the right. The solution is piecewise constant in most parts of the function, with occasional sharp jumps.

Theoretical results and explanations for the staircasing effect in the continuous ROF model can be found in [Caselles et al., 2008] and [Ring, 2000].



(a) Noisy image

(b) $\lambda = 0.1$

(c) $\lambda = 0.5$

(d) $\lambda = 1.0$

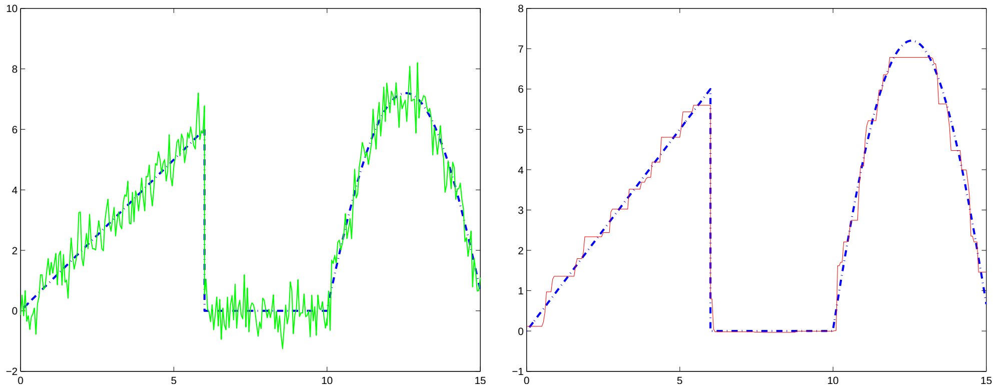
Figure 2.5: Reconstruction of a noisy image using the ROF model for different values of λ.

From a probabilistic point of view, the staircasing effect can be attributed to the use of the Laplacian prior together with MAP estimation, enforcing sparse solutions with respect to the image gradients. To remedy this effect, one has two choices in the construction of a model. First, one could replace the MAP procedure by a better estimator (which is the focus of this thesis). Second, one could regularize the model with a regularization term that better captures the statistics of the image (see for example [Bredies et al., 2009], [Chan et al., 2005], [Chan et al., 2007]).

2.3 Chapter Summary

In this chapter, we introduced a simple variational denoising model along with an algorithm to solve the model. While the model itself is very simplistic, its derivation nonetheless highlights the connection to Bayesian modeling and probabilistic methods.

In fact, a lot of models in Computer Vision are based on an optimization problem similar to (2.9), i.e. they can be formulated as an energy minimization problem $\min_u E(u)$. Consequently, such energies can be treated in the context of probability density functions, inducing a posterior probability density that is



(a) Noisy signal (b) Denoised signal

Figure 2.6: Reconstruction of a one-dimensional signal using the ROF model. The staircasing effect is clearly visible at the slopes of the function.

in general given by

$$p(u|.) = \frac{1}{Z} e^{-\frac{E(u)}{T}}$$

This allows us to use methods from stochastic estimation and Bayesian inference to solve the optimization problem as well as to leverage the result by estimating other quantities than the MAP. The rest of this thesis will be concerned with this probabilistic formulation and its advantages over the deterministic formulation.

Chapter 3

Estimation Theory

In the preceding chapter, a model for the reconstruction of noisy image was discussed. The proposed method employed a probabilistic argument to construct a posterior probability density $p(u|f)$, which assigns any possible image u a probability according to the proposed image and noise model. The question that arises is how to choose an image from this distribution that optimally accounts for the model.

A very intuitive approach was already discussed: By choosing the image that maximizes the probability $p(u|f)$, we expect to get a good result (provided the model was well-designed beforehand). Formally, we look for the image u^* that satisfies

$$u^* = \arg\max_u p(u|f)$$

This procedure is called Maximum-A-Posteriori estimation (MAP), ultimately leading to a deterministic formulation of the initially stochastic problem. While it is somewhat intuitive to choose the image with the highest probability in the context of the model, there actually may exist images that would be characterized as better solutions that have a lower probability.

In [Nikolova, 2007] it has been shown that MAP estimation has some severe shortcomings, even for "perfect" models. MAP estimators may, depending on the actual form of the posterior, introduce distortions:

1. For multimodal posteriors, the MAP estimate finds the mode with the largest peak, not taking into account the probability mass around the estimate.
2. The resulting estimate does not follow the underlying model in most cases (in a statistical sense).

This chapter is concerned with the analysis of these problems as well as with the introduction of an estimator that is able to eliminate these problems.

3.1 MAP-Distortions in High Dimensions

It was already shown in the previous section that the MAP estimate depends only on the height of the maximal mode, not on the probability mass inside the mode. This behavior is especially problematic when the distribution is high-dimensional. High-dimensional distributions tend to concentrate their mass in a small area of their domain.

Katafygiotis and Zuev [2008] provide a notable example for such a concentration:

Consider the N-dimensional random vector $X = (X_1, \dots, X_n)^T$ where each component is independent and identically distributed according to a standard Normal distribution, i.e. $X_i \sim \mathcal{N}(0, 1)$, for all $0 < i \leq N$.

Then the Euclidean distance R of the random variable X to the origin is given by

$$R = \sqrt{\langle X, X \rangle} = \sqrt{\sum_{i=1}^{N} X_i^2} \tag{3.1}$$

Note that the squared Euclidean distance R^2 leads to the definition of the Chi-square distribution with N degrees of freedom, i.e. $R^2 \sim \chi_N^2$.

Canal [2005] and Fisher [1922] show that $\sqrt{2R}$ can be approximated by a normal distribution with mean $\sqrt{2N-1}$ and unit variance, as N tends to infinity. A simple transformation shows that the Euclidean distance to the origin is distributed according to

$$R \sim \mathcal{N}(\sqrt{N - 1/2}, 1/2) \approx \mathcal{N}(\sqrt{N}, 1/2)$$

This is a remarkable result as it states that a large amount of the total probability mass is concentrated in a spherical ring around the origin, while the mode with maximal probability lies at the origin. As the number of dimensions rises, the sphere is pushed farther away from the origin. This implies that the MAP estimate is a relatively uncommon realization of the underlying probability distribution.

The concentration of mass phenomenon can be observed in more complex models as well, which is a hint that the MAP estimator may not be optimal in high-dimensional problems.

3.2 Distributions of MAP estimates

The previous chapter showed the emphasis on the modeling aspect in the design of variational models. Starting from assumptions on the statistical properties of the noise and some prior knowledge that was obtained from empirical observations, one derives a model that faithfully captures this information.

It seems to be a natural demand for an estimator that its estimates area distributed according to the given model. For example estimated u, is expected to be distributed according to the prior distribution (i.e. Laplacian of its gradients for the ROF model) while $u - f$ should look like the noise distribution (i.e. i.i.d. Gaussian). This rarely the case for MAP, however, which is another weakness of this type of estimator.

Numerical examples and analytic results of such distortions in discrete models are provided by Nikolova in [Nikolova, 2007]. Let us briefly recall the main results.

Consider the discrete one-dimensional ROF model

$$ROF(u; f, \lambda) = \|u - f\|_2^2 + \lambda \sum_{i=1}^{N} |u_i - u_{i+1}| \tag{3.2}$$

The single-pixel differences are distributed according to the pdf

$$f_{\nabla U_i}(t) = \frac{\lambda}{2} \exp\{-\lambda |t|\} \tag{3.3}$$

and independent for all $1 \leq i \leq N$. The random variable ∇U_i is given by $\nabla U_i = U_i - U_{i+1}$. We assume that $U_i \in \mathbb{R}$, therefore the pdf in (3.3) is continuous.

It was already noted that the ROF model suffers from the staircasing effect. It was not clear, whether this effect should be attributed to the model (i.e. the Laplacian prior) or to the MAP estimation. Using (3.3), the probability that the values of two neighboring pixels are equal, according to the prior model, is given by:

$$Pr(U_i = U_{i+1}) = Pr(\nabla U_i = 0) = 0 \tag{3.4}$$

The result comes from the continuity of the pdf and the fact that $\nabla U_i = 0$ constitutes a single probability event, which always has zero probability in a continuous probability space. This shows that the prior in principle does not favor piecewise constant regions.

Nikolova [2007], however, proves that the probability that two neighboring pixels are equal in the MAP estimate is non-zero . Let us briefly sketch the idea behind the proof:

Given the set J of points where the finite differences of the strict local minimizer u^* are zero, i.e.

$$J = \left\{ i \in 1, \ldots, N : u_i^* - u_{i+1}^* = 0 \right\}$$

and the set K_J of signals u which have finite differences equal to zero at the same points

$$K_J = \{ u \in \mathbb{R} : u_i - u_{i+1} = 0, \forall i \in J \}$$

[Nikolova, 2000] and [Nikolova, 2004] shows that J is nonempty and there exists an open neighborhood O_J of f where for each $\hat{f} \in O_J$, the corresponding minimizer $\hat{u}^*$ also exhibits finite differences equal to zero at the same points , i.e. $\hat{u}^* \in K_J$. Now note that a MAP estimate given $y \in O_J$ results in solutions that are actually a subset of K_J. This establishes the relation

$$Pr(U^* \in K_J) \geq Pr(F \in O_J) = \int_{O_J} \left(\int_{\mathbb{R}^N} p(f|u) p(u) du \right) df > 0$$

which is a contradiction to the prior probability (3.4) because it shows that

$$Pr(U^* \in K_J) = Pr(U_i^* = U_{i+1}^*, \forall i \in K_J) > 0$$

Therefore, the distribution of the MAP estimate is not identical to the prior model. This result is not only applicable for this particular one-dimensional example. In fact, it holds for any prior that is non-smooth at zero in the context of MAP estimation.

The result is remarkable as it states that no matter how well a model fits the underlying data, there is no hope of recovering a signal that fits the model using a MAP estimator. As long as one resorts to MAP estimates, there will always be staircasing (or similar effects if priors based on higher-order derivatives are used).

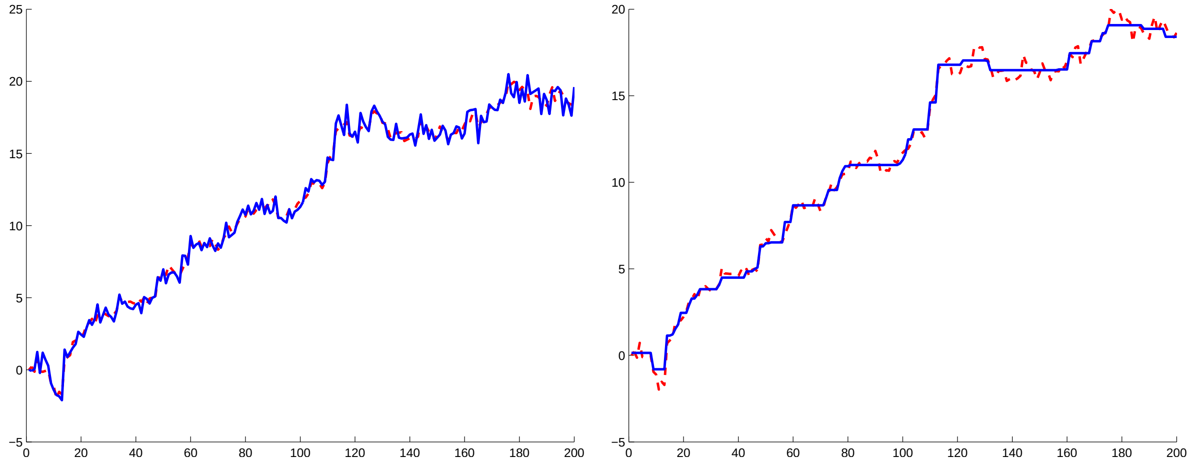
Let us consider a numerical example to back up the theoretical results:

We construct a Laplacian random walk by $x_{i+1} = x_i + L$ where L is distributed according to a Laplacian distribution with zero mean and variance β. It is easy to see that the differences $x_{i+1} - x_i$ are distributed according to a Laplacian prior as well.

If i.i.d. Gaussian noise with zero mean and variance σ^2 is added to a realization of the random walk, we get a noisy signal that perfectly fits the initial assumptions that led to the design of the ROF model. It is even possible to calculate the regularization parameter which is simply given by $\lambda = \frac{2\sigma^2}{\beta}$.

Figure 3.1(a) shows a single realization of such a random walk (the dashed line shows the true signal, whereas the solid line depicts the noisy signal). Figure 3.1(b) shows the reconstruction (solid), obtained using Algorithm 1, superimposed with the original signal (dashed). Even though the ROF model is able to perfectly model the signal, the MAP reconstruction still exhibits staircasing.

Figure 3.2 shows the empirical distribution of the differences of the true signal (3.2(a)) and the respective distribution of the MAP reconstruction (3.2(b)). The reconstruction is strongly peaked at zero, reflecting the many piecewise constant regions of the reconstruction. Figures 3.2(c) and 3.2(d) show the empirical distributions of the true noise and the residuals of the MAP reconstruction. The residuals are not Gaussian at all. Instead, with a strong peak at zero and the approximately exponential decay of the tails, it resembles a Laplacian, leading [Nikolova, 2007] to the remarkable observation that this model is better suited to remove impulse noise than Gaussian noise.



(a) A single realization of a Laplacian random walk with superimposed Gaussian noise. Dashed: Original signal. Solid: Noisy signal

(b) Reconstruction using MAP of ROF model. Dashed: Original signal. Solid: MAP reconstruction

Figure 3.1: Reconstruction of a signal obtained from a noisy Laplacian random walk ($\sigma^2 = 0.5$, $\lambda = 2$). Even if the signal perfectly fits the model, MAP fails to provide a faithful reconstruction

3.3 Bayesian Risk

The framework of Bayesian estimation provides us with tools to develop and analyze estimators based on the risk of taking a wrong estimate. By introducing loss functions, it is possible to associate a “cost” with an estimate, reflecting the regret of being a wrong estimate.

Consider the true image u' and an estimate u (we do not consider how this estimate was obtained in the first place). Then a loss function $L : \mathbb{R}^{|\Omega|}$ x $\mathbb{R}^{|\Omega|} \to \mathbb{R}$. can be used to express one’s needs for the reliability of an estimator.

Considering that the true value u' is unknown in general (if u' is known, the estimator u is obsolete in the first place) and that we are dealing with probabilistic measures, it is necessary to assign some probabilistic measure to the loss function as well to make this notion useful. This can be achieved using the concept of Bayesian Risk.

The Bayesian Risk $R(u)$ associated with a loss function $L(u, u')$, is defined as the expected value of the loss function, with respect to the posterior distribution of the quantity of interest:

$$R(u) = \mathbb{E}_{u'|f}\{L(u, u')\} = \int_{\mathbb{R}^{|\Omega|}} L(u, u')p(u'|f)du'$$

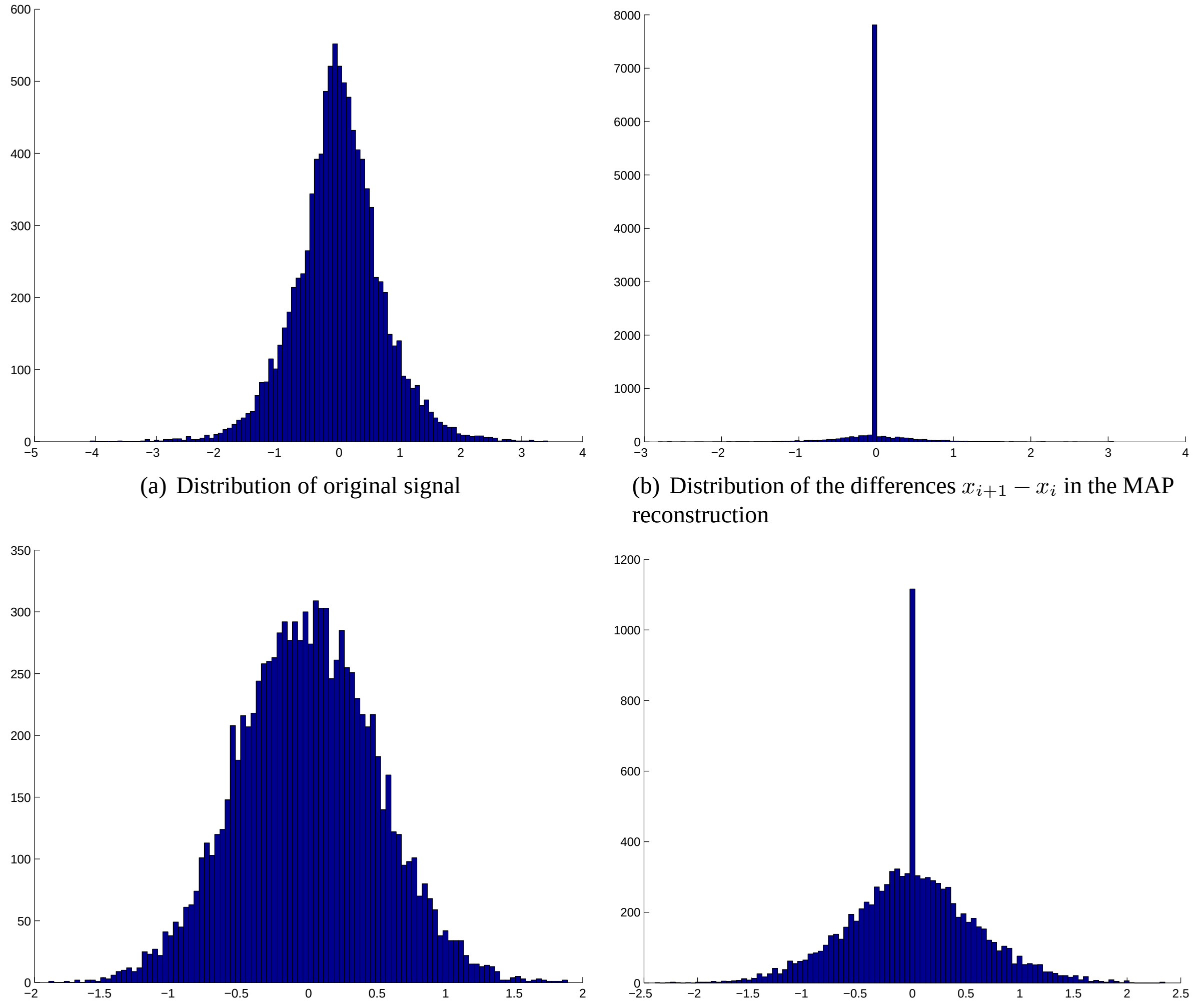
By minimizing the Bayesian Risk, it is possible to derive an estimator that is optimal with respect to a given loss function and posterior distribution. Such estimators are called Bayesian estimators.

Formally, a Bayesian estimator u^* satisfies

$$u^* = \arg\min_u R(u) \tag{3.5}$$

for some risk $R(u)$.

Note that the definition of the Risk function gives rise to other (non-bayesian) estimators as well. One popular example thereof is the so-called minimax estimator that minimizes the risk with respect to the least favorable distribution $p(u|f)$. This can be interpreted as the estimator that performs best in the worst case of input data (i.e. the input data does not match the model).



(a) Distribution of original signal

(b) Distribution of the differences $x_{i+1} - x_i$ in the MAP reconstruction

(c) Distribution of the noise

(d) Distribution of the residuals

Figure 3.2: True distributions (left) and distributions of MAP reconstruction (right). Neither the the distribution of the differences $x_{i+1} - x_i$ nor the residuals follows the original distributions.

A common and simple loss function is given by the hit-or-miss loss, which assigns the same cost to every wrong estimate:

$$L_0(u, u') = \begin{cases} 0 & \text{if } u = u' \\ 1 & \text{else} \end{cases} \tag{3.6}$$

For this type of loss, the Bayesian estimator is given by:

$$\begin{aligned} u^* &= \arg\min_u \left\{ \mathbb{E}_{u'|f}\{L_0(u, u')\} \right\} = \arg\min_u \left\{ \int_{\mathbb{R}^{|\Omega|}} L_0(u, u') p(u'|f) du' \right\} \\ &= \arg\min_u \left\{ \int_{u \neq u'} p(u'|f) du' \right\} = \arg\min_u \left\{ 1 - \int_{u = u'} p(u'|f) du' \right\} \\ &= \arg\min_u \left\{ -\delta(u, u') p(u'|f) \right\} = \arg\max_u \left\{ p(u|f) \right\} \end{aligned} \tag{3.7}$$

where $\delta(u, u')$ denotes the dirac function, i.e.

$$\delta(u, u') = \begin{cases} \infty & \text{if } u = u' \\ 0 & \text{if } u \neq u' \end{cases}$$

(3.7) shows that the estimator minimizing the hit-or-miss loss is given by the MAP estimate. This derivation also shows that the MAP estimate is a point estimate (see the last line, involving diracs delta) that inevitably fails to capture the information that is present in the posterior distribution.

Another popular choice for a loss function is given by the squared-error loss:

$$L_2(u, u') = \|u - u'\|^2 \tag{3.8}$$

Unlike the hit-or-miss loss the squared-error loss assigns different "cost" to wrong estimates, depending on the Euclidean distance to the true value. The Bayesian estimator for this kind of loss function is given by

$$u^* = \arg\min_u R(u) = \arg\min_u \left\{ \int_{\mathbb{R}^{|\Omega|}} \|u - u'\|^2 p(u'|f) du' \right\} \tag{3.9}$$

and the minimum of this function is attained where

$$\frac{\delta R(u)}{\delta u} = 2 \left(u \int_{R^\Omega} p(u'|f) du' - \int_{R^\Omega} u' p(u'|f) du' \right) = 0$$

which finally leads to

$$u^* = \int_{R^\Omega} u' p(u'|f) = \mathbb{E}_{u|f}\{u\}$$

i.e. the estimator minimizing the Bayesian risk defined by the squared-error loss is given by the expected value of the posterior. We will further call this estimator the Least-Squares estimator, or short LSE. By definition, the expected value is a summary statistic, effectively compressing the whole knowledge about the posterior into a single estimate. Depending on the posterior distribution, such a summary may yield better results than a single point estimate. Note that other, more sophisticated, loss functions are possible (see for example [Rue and Hurn, 1997]), we will, however, focus on the LSE estimate.

The estimation of such a summary statistics is much more difficult than a simple point estimate. One has to have complete knowledge of all possible outcomes of the underlying probability experiment to exactly infer the estimate (as the integral in the expected value is taken over the whole problem domain).

Given typical problem spaces (consider a discrete binary image as small as 64 by 64 pixels, leading to $(64\text{x}64)^2 \approx 17\text{x}10^6$ possible image configurations), it is not possible to exactly infer the expected value. Instead, one has to rely on approximations. The next chapter will focus on such approximations.

Let us first turn to an evaluation of the two estimators. Bayesian estimation theory features some notions to characterize the performance of an estimator. One common measure is the mean-bias of an estimator, defined as

$$\text{Bias}(u) = \mathbb{E}\{u\} - u' = \mathbb{E}\{u - u'\}$$

An estimator with $\text{Bias}(u) = 0$ is called unbiased. Loosely speaking one could say that an unbiased estimator will, on average, yield the correct estimate. It is neither necessary for an estimator to be unbiased nor does it automatically guarantee that it is the best among all estimators. Seeking for an unbiased estimator is however a common starting point in the design phase.

Another important performance measure is given by the variance of an estimator:

$$\text{Var}(u) = \mathbb{E}\{\|u - \mathbb{E}\{u\}\|^2\}$$

The variance of an estimator essentially captures how far the estimates are from the mean on average.

Last, we define the mean-squared error of an estimator as

$$\text{MSE}(u) = \mathbb{E}\{\|u - u'\|^2\} = \text{Var}(u) - \|\text{Bias}(u)\|^2$$

These measures are difficult to derive for the MAP estimator, as the relation of the data to the estimate are only implicitly given (note that for certain functionals, one can derive an approximation even for implicit relations [Fessler, 1996]). We can, however, deduce some conclusions for the MAP estimate based on the performance analysis of the LSE estimator:

The Bias of the LSE estimator is given via the law of iterated expectations:

$$\text{Bias}(u_{LSE}) = \mathbb{E}\{u_{LSE}\} - u' = \mathbb{E}\{\mathbb{E}\{u'|f\}\} - u' = u' - u' = 0$$

u_{LSE} is therefore an unbiased estimator. With any unbiased estimator, we have that

$$\text{MSE}(u_{LSE}) = \text{Var}(u_{LSE})$$

Recall that the LSE estimator minimizes the expected value of the squared-error loss function, which coincides with the mean-squared error. This implies that the LSE estimator is the estimator with minimal variance among all unbiased estimators.

For the analysis of the MAP estimate, we can therefore conclude that

- if the MAP estimate coincides with with the LSE estimate, it is also unbiased and minimum variance. This case only happens if the posterior is both unimodal and symmetric.
- if, in contrast, the MAP estimate is different from the LSE estimate, it is biased. This is the general and more interesting case in the context of variational methods.

To develop some intuition on when and why the LSE estimate can yield better outcomes than the MAP, let us analyze some examples. Figure 3.3 shows simple one-dimensional examples for the outcome of a MAP estimation (red) versus a LSE estimation (blue). For distributions that are symmetric and unimodal (Figure 3.3(a)), the MAP and LSE estimators yield the same result (as the center of mass and the maximal mode are located at the same place). For unimodal non-symmetric distributions, the LSE estimator is shifted towards the heavier tail of the distribution (as depicted in Figure 3.3(b)).

For multimodal distributions, the actual location depends on the distribution of the modes. Figure 3.3(c) shows an example where the LSE estimator would clearly outperform the MAP estimate. While MAP is located at the strongest mode (that actually captures little probability mass), the LSE estimate is located near the peak of the mode with much larger probability mass (that has a nearly as large extremal value as the MAP peak), giving much more support for this estimate in the context of the original model.

It should be noted that for multimodal posteriors, one cannot generally hope that the LSE estimate outperforms the MAP estimate. Figure 3.3 shows an example where the expected value of the distribution lies in between two strong modes. This estimate has a very low probability, which would make this choice a risky one. The MAP estimate, however, has to arbitrarily choose from the 2 modes, as both have equal probability.

3.4 Chapter Summary

This chapter tried to highlight problems that arise within the framework of MAP estimation. Using Bayesian estimation theory, we developed an alternative estimator that in theory yields better results than the MAP estimate.

Let us briefly summarize the insights gained so far:

- MAP estimation suffers from severe distortions whenever non-smooth priors are present in the functional, as it is the case in most variational models in Computer Vision.
- The LSE estimator may yield better results depending on the shape of the posterior distribution. Especially if the posterior is asymmetric and unimodal, it can be expected that a reconstruction by the LSE estimator is superior to the MAP estimate.

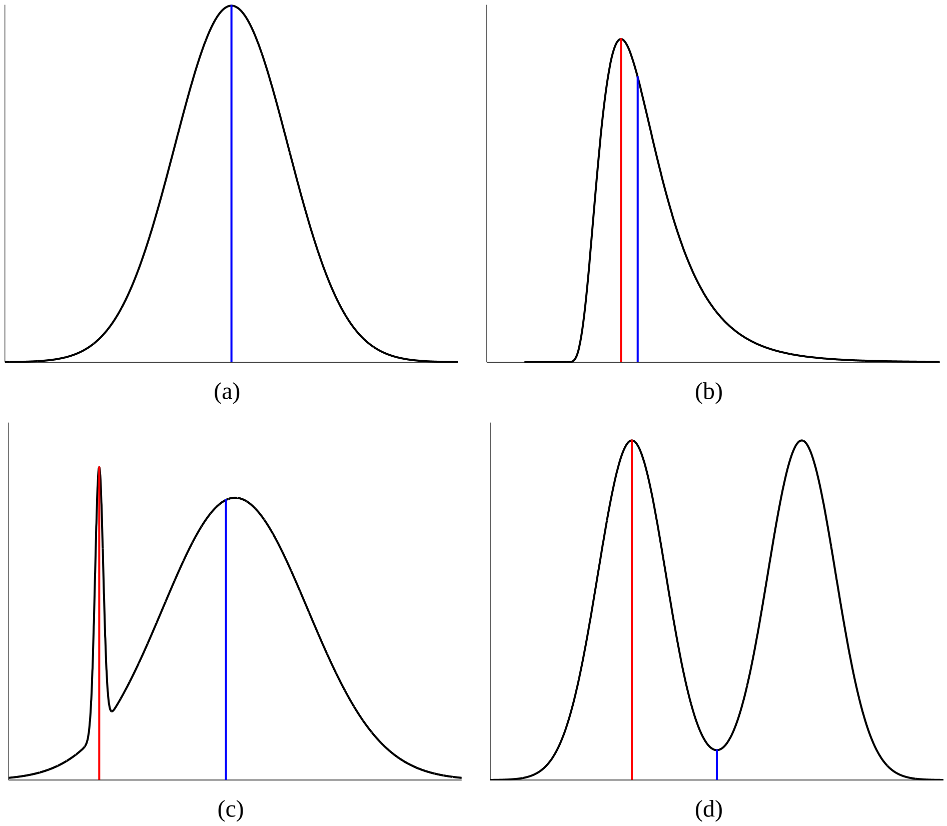


Figure 3.3: MAP (red) and LSE (blue) estimates for different probability densities. (a) Shows an unimodal, symmetric distribution. The maximum and the center of mass coincide, therefore MAP and LSE yield the same results. (b) Asymmetric, unimodal distribution. The LSE is shifted towards the heavy tail of the distribution. (c) Bimodal distribution. While MAP is located within the mode with highest probability, the LSE is located near the peak of the mode with greater probability mass. (d) Symmetric, bimodal distribution. The LSE estimate has very low probability. The MAP estimator has to arbitrarily choose between the two modes as both have equal probability.

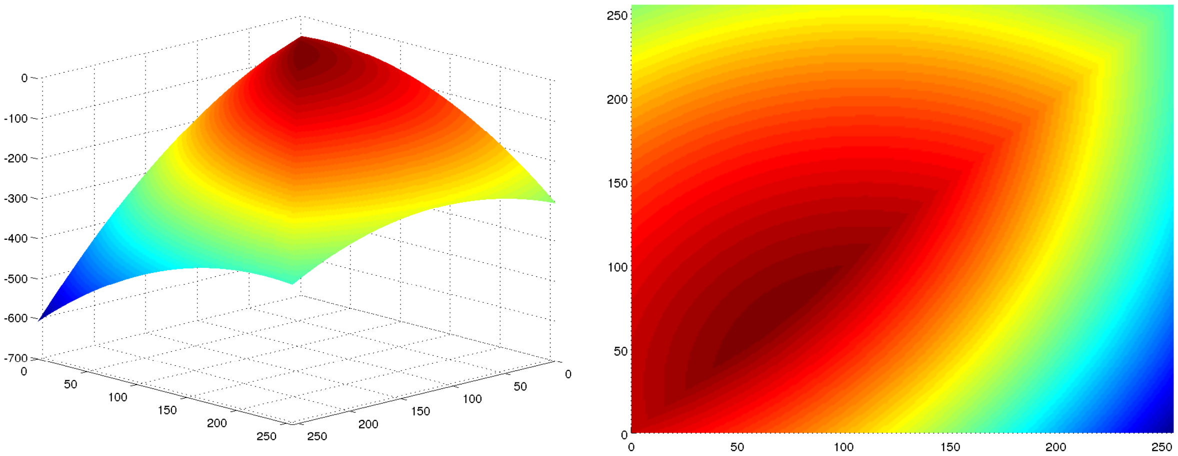
The ROF model is convex, which implies that the posterior is log-concave and therefore by definition unimodal. Furthermore, it is asymmetric in general (the pdf is symmetric if f is constant everywhere in Ω). The ROF model is hence in principle well-suited for an LSE estimator. Figure 3.4 visualizes the shape of the distribution for a simple example involving only two pixels.

The Tikhonov model on the other hand defines a unimodal symmetric pdf, as depicted in Figure 3.5. The LSE and the MAP coincide in this model and a modified estimation procedure therefore cannot enhance the results.

Recall that the LSE estimator is given by

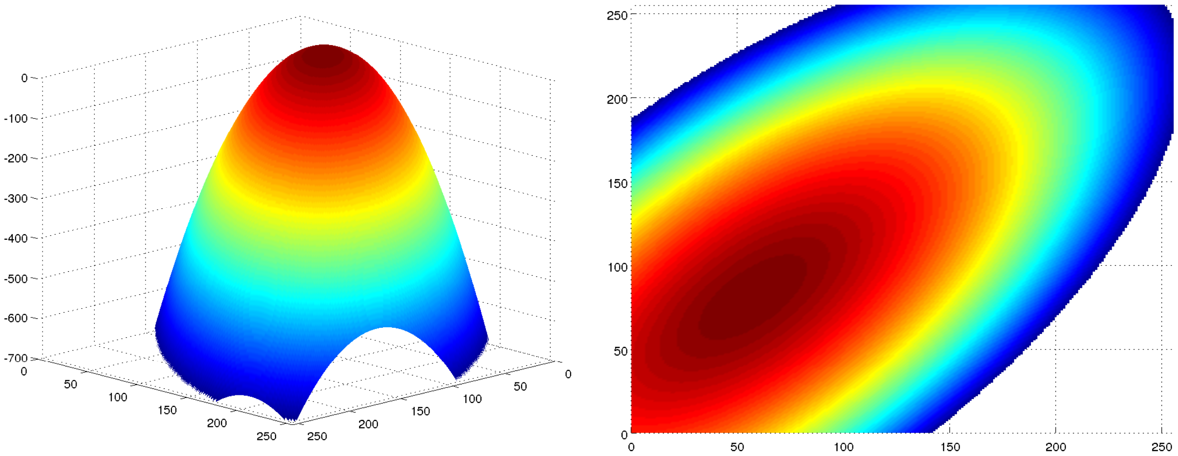
$$u^* = \int_{\mathbb{R}^{|\Omega|}} up(u|f)du \tag{3.10}$$

The integral in (3.10) has to be taken over every possible image configuration of a given size $|\Omega|$. It is clearly impossible to exactly solve this integration, even for relatively small images. The following chapter is therefore devoted to the development of a procedure that allows us to approximately solve (3.10).



(a) (b)

Figure 3.4: Logarithmic plot of the pdf of the ROF model for two adjacent pixels. The possible values of a pixel was constrained to range [0,255] and the variance of the prior and noise distribution was set to $\beta = 1$ and $\sigma^2 = 100$ respectively. The two "noisy" pixels where set to the values $f_1 = 10$ and $f_2 = 128$. The resulting MAP reconstruction yields $u_1 = u_2 = 69$, whereas the LSE reconstruction yields $u_1 = 68.19$ and $u_2 = 68.80$.



(a) (b)

Figure 3.5: Logarithmic plot of the pdf of the Tikhonov model for two adjacent pixels. The possible values of a pixel was constrained to range [0,255] and the variance of the prior and noise distribution was set to $\beta = 50$ and $\sigma^2 = 50$ respectively. The two "noisy" pixels where set to the values $f_1 = 10$ and $f_2 = 128$. The resulting pdf is symmetric. The LSE and the MAP reconstruction yield $u_1 = 57$ and $u_2 = 81$.

Chapter 4

Markov Chain Monte Carlo

The estimator (3.10) poses an extremely high-dimensional integral. Deterministic numerical integration methods, such as the Newton-Cotes Formula or Simpson's Rule, fail to approximate the integral in such a scenario. Problems that involve high-dimensional integration arise in a variety of fields, including physics, finance, statistics and computational biology. The development of efficient methods to numerically solve such problems has therefore become an important research topic in mathematics (and especially statistics).

To exemplify the shortcomings of deterministic numerical integration for high-dimensional problems, we consider a simple example based on the rectangle rule [Arnold, 2001]:

Given a function $f : [a, b] \rightarrow \mathbb{R}$ the area

$$I = \int_a^b f(x)dx$$

is approximated by partitioning the interval $[a, b]$ in N equidistant sub-intervals of width $h = \frac{b-a}{N}$:

$$\int_a^b f(x)dx \approx \sum_{i=0}^{N} f(a + hi)h$$

where $a + Ni = b$.
Assuming that $f(x)$ is continuously differentiable on $[a, b]$, the approximation error is given by

$$E = \frac{f'(\xi)}{2N}(b-a)^2 \tag{4.1}$$

for some $\xi \in [a, b]$. (4.1) states that the error linearly decreases as the number of sub-intervals N grows. By denoting $O(\frac{1}{N})$ as the error term, the exact integral then reads

$$\int_a^b f(x)dx = \sum_{i=0}^{N} f(a + hi)h + O(\frac{1}{N})$$

Extension of one-dimensional numerical integration methods to multi-dimensional integrals is straight forward:

Assume a function is defined on the d-dimensional hypercube $C : [a, b]^d$, i.e. $f : C \rightarrow \mathbb{R}$, then the integral over this hypercube is given by:

$$\int_C f(\mathbf{x})d\mathbf{x} = \int_a^b \dots \int_a^b f(\mathbf{x})dx_1 \dots dx_d$$

and repeated application of the rectangle rule yields

$$\int_C f(\mathbf{x})d\mathbf{x} \approx \sum_{i_1=0}^{N-1} \cdots \sum_{i_d=0}^{N-1} f(a+hi_1,\ldots,a+hi_d)h \tag{4.2}$$

The integrand in (4.2) is evaluated $n = (N+1)^d$ times, and since the error depends on the total number of evaluations, the approximation error is of order $O(n^{-1/d})$. Thus, the error bound gets exponentially worse with a rising number of dimensions. While the error may be substantially smaller for more sophisticated integration rules in the one-dimensional case, the statement that the error grows exponentially with the number of dimensions holds for all deterministic numerical integration methods.

The most common and general tool to tackle high-dimensional integration problems is known as the Monte Carlo method. The term “Monte Carlo” refers to a class of algorithms that use repeated random sampling of the possibly very large input domain (the integration domain in case of Monte Carlo integration) to generate an output.

A multi-dimensional integral

$$I = \int_\Omega f(\mathbf{x})d\mathbf{x}$$

is approximated by randomly generating a set of N sample points $\{\hat{\mathbf{x}}_1,\ldots,\hat{\mathbf{x}}_N\},\ \hat{\mathbf{x}}_i \in \Omega$ by uniform sampling under the integration domain and evaluating

$$\hat{I} = \frac{1}{N}\sum_{i=1}^{N} f(\hat{\mathbf{x}}_i)$$

Provided that sampling is done correctly (the whole integration domain is covered, no statistical pathologies from pseudo-random number generators), the law of large numbers states that

$$\lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} f(\hat{\mathbf{x}}_i) = I$$

While it is not possible to give a deterministic error bound for Monte Carlo integration, a probabilistic error bound can be obtained from the Central Limit Theorem:

$$\lim_{N\to\infty} Pr\left(\left|\frac{1}{N}\sum_{i=1}^{N} f(x_n) - I\right| \leq z\frac{\sigma(f)}{\sqrt{N}}\right) = \frac{1}{\sqrt{2\pi}}\int_{-z}^{z} \exp -\frac{t}{2}dt$$

This is a remarkable result as it states that the error is independent of the number of dimensions. While this justifies the use of Monte Carlo methods for high-dimensional integration, a convergence rate of $\frac{1}{\sqrt{N}}$ is, however, relatively slow (to half the error for a given N, one has to quadruple the number of samples). Markov Chain Monte Carlo (MCMC) algorithms try to mitigate this slow rate of convergence by exploiting the concentration of mass phenomenon that was already discussed in the preceding chapter. Instead of uniform sampling under the integration domain and weighting the samples using the integrand, MCMC algorithms sample areas that have a high contribution to the integral more densely than areas with low contribution and do a uniform weighting of the samples.

The remainder of this chapter is organized as follows:

Section 4.1 introduces Markov Chains that are defined on uncountable state spaces and introduces the most important properties of such chains that are needed for the construction of MCMC algorithms.

Section 4.2 introduces the two most important algorithms in MCMC theory, the Gibbs sampler and the Metropolis-Hastings algorithm, and combines ideas from both algorithms to a sampling procedure, which is particularly appropriate for image processing tasks.

A procedure that allows the presented samplers to quickly converge without manual tuning of parameters, is introduced in section 4.3.

4.1 Markov Chains

Let $X_t \in S$ denote the realization of a random variable defined on the state space S, at time t, i.e. the set $\{X_t\}$ defines a random process. We focus on discrete-time Markov chains, where t is an integer larger than zero. A random process $\{X_t\}$ is called a Markov Chain of order k, if the probability of visiting the future state X_{t+1} given the current state X_t does not depend on its past states $X_{t-n}, \quad n = k, \ldots, t$ for some $k > 0$.

Formally, this can be written as:

$$Pr(X_t = s_j | X_0 = s_0, \ldots, X_{t-1} = s_i) = Pr(X_t = s_t | X_{t-k} = s_l, \ldots, X_{t-1} = s_i) \tag{4.3}$$

The number of past states the chain depends on is called the order of the chain. For the remainder of this thesis, we stick to first-order Markov Chains, i.e. chains where the probability of visiting a particular next state only depends on the current state. (4.3) then simplifies to:

$$P(i,j) = Pr(X_t = s_j | X_0 = s_0, \ldots, X_{t-1} = s_i) = Pr(X_t = s_j | X_{t-1} = s_i) \tag{4.4}$$

$P(i,j)$, the probability of moving to state s_j given state s_i, is called the transition kernel of the chain.

This definition assumes a finite (i.e. countable) state space S. While any implementation of variational models on a digital computer clearly involves finite state-spaces, the analysis of such models is typically carried out in uncountable spaces. It is therefore advantageous to apply MCMC theory in an uncountable state-space as well, so that no assumption on discretization of the underlying variational model has to be made.

A straight-forward generalization of first-order Markov chains to uncountable state-spaces (further referred to as general state-spaces) is given by:

$$P(x, A) = Pr(X_t \in A | X_{t-1} = x), \qquad A \in \mathbb{B}(S) \tag{4.5}$$

where $\mathbb{B}(S)$ denotes the Borel set of S.

The transition kernel $P(x, A)$ can be interpreted as the probability density function of moving to subspace A from the current state x.

Note that the transition kernel can in principle be a function of the evolution time t. We limit the kernel to be time invariant for now:

$$Pr(X_t \in A | X_{t-1} = x) = Pr(X_{t+m} \in A | X_{t+m-1}), \qquad \forall m$$

Such a chain is also called time-homogeneous or stationary.

The evolution of $\{X_t\}$ is fully governed by its transition kernel and its initialization X_0. Using induction, the probability that $X_t \in A$ after n steps when starting at $X_0 = x$ can be calculated recursively:

$$\begin{aligned}
P(x, A)^{(0)} &= \delta_x(A) \\
P(x, A)^{(1)} &= \int_S P(x, dy) \delta_y(A) = P(x, A) \\
P(x, A)^{(2)} &= \int_S P(x, dy) \{ \int_S P(y, dz) \delta_z(A) \} = \int_s P(x, dy) P(y, A)^{(1)} \\
&\vdots \\
P(x, A)^{(n)} &= \int_S P(x, dy) P(y, A)^{(n-1)}, \quad x \in S, A \in \mathbb{B}(S)
\end{aligned} \tag{4.6}$$

where $\delta_x(A) = 1$ if $x \in A$ and 0 otherwise.

Markov Chains that fulfill certain regularity conditions eventually converge to a limiting distribution $\pi^*(.)$. MCMC algorithms exploit this fact by cleverly constructing a chain, so that its limiting distribution coincides with a desired target distribution. Definition 2 introduces two important concepts for such a construction:

Definition 2. *[Meyn and Tweedie, 1993] Let $\{X_t\}$ be a Markov chain, defined on the general state-space S. We call $\{X_t\}$ ϕ-irreducible with respect to a measure ϕ, if for all $A \in \mathbb{B}(S)$*

$$\phi(A) > 0 \Rightarrow P(x, A)^{(n)} > 0, \qquad \forall x \in S$$

holds.

Furthermore, assume that a chain is ϕ-irreducible. Let us denote by

$$\tau_A = \sum_{n=1}^{\infty} \mathbf{1}\{X_t \in A\}$$

the number of times the chain visits the set A. Then a chain is said to be Harris-recurrent, if

$$Pr(\tau_A = \infty) = 1$$

for all sets $A \subseteq S$. Such a chain visits every state infinitely often, independent of the initialization.

The concept of ϕ-irreducibility states that each relevant state with respect to a distribution ϕ is visited with non-zero probability, regardless of the initial value. Recurrence on the other guarantees that every state is visited infinitely often. Any chain that is ϕ-irreducible and Harris-recurrent has a unique invariant distribution π. Moreover, this stationary distribution coincides with ϕ. Keeping in mind that the goal of MCMC algorithms is to construct a chain with target invariant distribution π, such a chain must therefore be π-irreducibly as well as Harris-recurrent.

Using these conditions, we are finally able to define the invariant distribution:

Definition 3. *Let π be a probability density function defined on S and $\{X_t\}$ be the time-homogeneous,π-irreducible, Harris-recurrent Markov chain with transition kernel $P(.,.)$, defined on the state-space S. We call π the invariant distribution of $\{X_t\}$, if π satisfies*

$$\lim_{n\to\infty} P(x, A)^{(n)} = \pi(A), \qquad \forall A \in \mathbb{B}(S)$$

Definition 3 implies that the invariant distribution also satisfies:

$$\pi(A) = \int_S P(x, A)\pi(x)dx, \qquad \forall A \in \mathbb{B}(S) \tag{4.7}$$

Relation (4.7) is a condition to test whether $\pi(.)$ is the invariant distribution of the chain with transition kernel $P(.,.)$. To simulate from a given target distribution, this relation has to be inverted: For a given $\pi(.)$, find a transition kernel $P(.,.)$ so that (4.7) holds. It turns out that explicit construction of such a transition kernel is difficult or even impossible (even for simple π).

A last restriction on the chain enables us to simulate a chain with a predetermined invariant distribution without explicit construction of the transition kernel:

$$\pi(A)P(A, x) = \pi(x)P(x, A) \tag{4.8}$$

(4.8) is called the detailed balance condition. It essentially states that the properties of the chain do not change if the chain is run backwards in time. Thus, chains that maintain detailed balance are also often called reversible.

Proposition 1. *Any Harris-recurrent, π-irreducible Markov chain with transition kernel $P(.,.)$ that fulfills condition* (4.8) *has $\pi(.)$ as its unique invariant distribution.*

Proof. Integrating both sites of (4.8) with respect to x yields

$$\pi(A) \int_S P(A,x)dx = \int_S \pi(x)P(x,A)dx$$

The integral on the left-hand site evaluates to one which shows the equivalence to (4.7). □

So far, we have seen what properties a Markov chain has to fulfill in order to have an invariant distribution. Recall that our initial motivation for the introduction of Markov Chains was to approximate a possibly very high-dimensional integral, by dense sampling of the integrand in areas with high contributions to the overall integral and sparser sampling in areas with low contributions. The following theorem theoretically justifies such a procedure:

Theorem 1. *[Meyn and Tweedie, 1993] Let $\{X_t\}$ be a Markov chain, defined on the general state-space S that has a unique invariant distribution π and is Harris-recurrent. Then for any function $g \in L^1$*

$$\lim_{n\to\infty} \frac{1}{N} \sum_{k=1}^{N} g(X_n) = \int_S g(u)\pi(u)du = \mathbb{E}\{g(u)\}$$

holds and we call the chain ergodic.

This theorem states that we can approximate an integral by uniformly weighting samples X_n, as long as those samples are distributed according to π. A useful fact is that any π-irreducible and Harris-recurrent chain is ergodic.

Proposition 1 together with (4.8) are the fundamental relations that are exploited by the algorithms in the following section to sample from a target distribution. Note, however, that detailed balance and therefore time-reversibility is a sufficient, not a necessary condition for the convergence of the chain to an invariant distribution.

4.2 Samplers

As already mentioned in the preceding section, the goal of MCMC algorithms is to construct a Markov Chain that has a desired invariant distribution $\pi(x)$. Once the chain is in its stationary regime, subsequent samples are distributed according to $\pi(x)$.

The probably most important sampling algorithms are the Metropolis algorithm [Metropolis et al., 1953], and its extension, the Metropolis-Hastings Algorithm [Hastings, 1970]. This algorithm has some very favorable properties. First, the algorithm does not impose any restrictions on the target distribution. This is specifically important for variational methods in image processing considering the sheer amount of different regularizers and fidelity terms. Second, the target distribution has to be known only up to a normalizing constant, and therefore needs no evaluation of the partition function.

The algorithm basically generates a potential new state based on some (arbitrary) proposal distribution. The proposed state is accepted to become the next state of the chain if it moves the chain uphill in the target distribution (i.e. to regions of higher probability). If this is not the case, the proposed state is still accepted with a certain probability that is always non-zero. After a certain number of iterations, the detailed balance condition is met and subsequent samples from the chain are distributed according to the desired target distribution.

Formally, we have that given a state x, a new state y is proposed according to some proposal distribution $q(y|x)$ (note that an explicit dependence on the current state x is not necessary, i.e. a proposal distribution could also be of the form $q(y|x) = q(y)$). The proposed state y is then accepted with probability

$$\alpha(x, y) = \min\left\{\frac{\pi(y)q(x|y)}{\pi(x)q(y|x)}, 1\right\}$$

and rejected with probability $1 - \alpha(x, y)$, and the algorithm proceeds with the same procedure from the new state.

Algorithm 2 Metropolis-Hastings Algorithm

```
Choose an initial state $X_0$
Choose a proposal distribution $q$
Set $k = 0$
loop {Metropolis-Hastings iteration}
    Draw a new potential state $X_{k+1/2} \sim q(X_{k+1/2}|X_k)$
    $\alpha \Leftarrow \frac{\pi(X_{k+1/2})q(X_k|X_{k+1/2})}{\pi(X_k)q(X_{k+1/2}|X_k)}$
    $X_{k+1} \Leftarrow X_{k+1/2}$ with probability $\min(\alpha, 1)$
    $X_{k+1} \Leftarrow X_k$ with probability $1 - \min(\alpha, 1)$
    $k \Leftarrow k + 1$
end loop
```

The basic iterations of the Metropolis-Hastings procedure are summarized in Algorithm 2. Note that each run of the loop in Algorithm 2 is called a Metropolis-Hastings iteration and generates one sample.

Proposition 2. *For an appropriate proposal distribution $q(y|x)$, Algorithm 2 converges to the target distribution $\pi(x)$ as $k \to \infty$*

Proof. First, consider the case $x \neq y$:

The transition kernel is then given by

$$P(x, y) = q(y|x)\alpha(x, y)$$

It is easy to see that the proposal distribution has to meet

$$\pi(A) > 0 \Rightarrow q(A|x)^{(n)} > 0, \qquad x \in S$$

to establish π-irreducibility, i.e. the proposal distribution is able to reach all relevant areas of the target distribution.

By substituting into (4.8), we get

$$\pi(x)q(y|x)\alpha(x, y) = \pi(y)q(x|y)\alpha(y, x)$$

Now assume $\alpha(x, y) < 1$, then it immediately follows that $\alpha(y, x) = 1$. Rearranging the equality then leads to

$$\alpha(x, y) = \frac{\pi(y)q(x|y)}{\pi(x)q(y|x)}$$

which is true by construction and establishes detailed balance. The case $\alpha(y, x) < 0$ follows analogously by symmetry.

The second case, the probability of remaining in a state, is given by

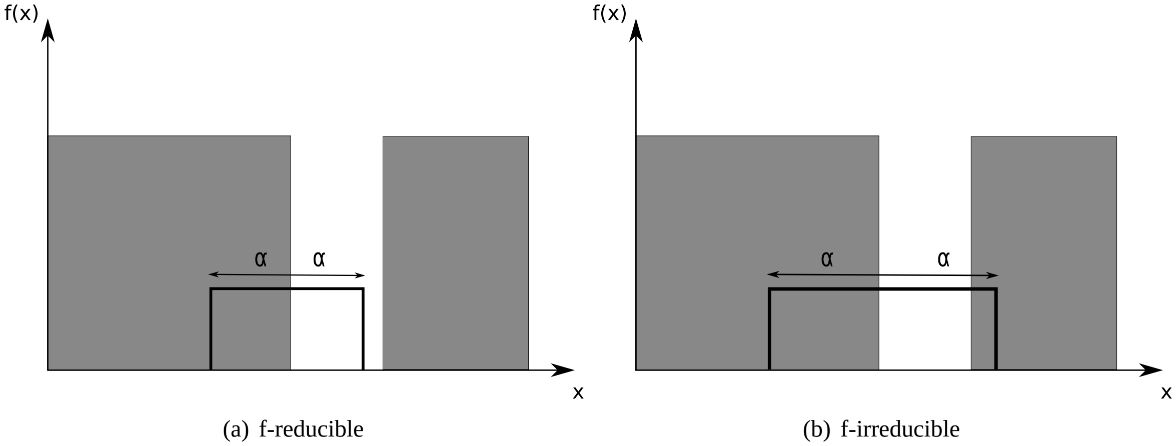
$$P(x, x) = 1 - \int_{S/\{x\}} q(y|x)\alpha(x, y)dy$$

The detailed balance condition is trivially met in this case. Summing up both cases, we see that the chain is reversible and π-reducible, which concludes the proof. □

The target distribution has to be known only up to a multiplicative normalizing constant in this algorithm because state transition probabilities only depend on the ratio of the target distribution evaluated at two states. This means that there is no need to evaluate the partition function $Z(f)$, which strongly enhances the applicability of this algorithm.

Let us now focus on the proposal distribution q:

A simple example of the influence of the proposal distribution $q(y|x)$ on π-irreducibility is depicted in Figure 4.1. Both graphs assume that the Metropolis-Hastings algorithm was initialized within the left segment of the target distribution $f(x)$. A chain will eventually reach the right boundary of this segment. Using a proposal distribution as shown in 4.1(a), the probability that the chain jumps to the other segment is zero, and the algorithm fails to correctly sample from the target distribution. Running the algorithm with a proposal distribution that allows larger jumps, as shown in 4.1(b), in contrast, results in an f-irreducible chain, which enables the correct sampling of the target.



(a) f-reducible

(b) f-irreducible

Figure 4.1: Influence of the proposal distribution on the convergence of Metropolis-Hastings algorithm. The area under $f(x)$ that is to be approximated is depicted in gray. (a) The proposals are too small. If the initialization lies within the left (larger) area, there is zero probability that the right area will be hit. Conversely, if one starts the sampling in the right area, the left area will never be hit. The chain therefore fails to sample the target $f(x)$ correctly. (b) The proposal distribution is allowed to propose jumps that are large enough so that there is a non-zero probability of hitting both areas, independent of the initialization. The chain is therefore f-irreducible and can correctly sample the target $f(x)$.

While the specific form of the proposal distribution can arbitrarily be chosen in principle (as long as π-irreducibility can be established with it), it should be subject to some considerations to allow a good performance of the algorithm. Typical runs of the algorithms involve hundreds of thousands of iterations and therefore a proposal distribution that is easy and fast to simulate should be chosen. By contrast the proposal distribution should be similar to the target distribution in order to explore the space of the target distribution in reasonable time. Typical choices for proposal distributions are the uniform distribution or the normal distribution, centered at the current state. Both distributions are fast to simulate. Furthermore they are symmetric, which simplifies the computation of the update probabilities to

$$\alpha \Leftarrow \frac{\pi(X_{k+1/2})q(X_k|X_{k+1/2})}{\pi(X_k)q(X_{k+1/2}|X_k)} = \frac{\pi(X_{k+1/2})}{\pi(X_k)} \tag{4.9}$$

We call the algorithm the Metropolis algorithm if the proposal distribution is symmetric, and the Metropolis-Hastings algorithm if said distribution is asymmetric.

4.2.1 Gibbs sampling

Another very popular sampler especially within the Computer Vision community is the Gibbs sampler, introduced by Geman and Geman [1993]. For this sampling procedure, one assumes that while it is infeasible to sample from the multivariate target distribution, it is easy to sample from the univariate conditional distribution of a single component (where the distribution is conditioned on all remaining components).

The Gibbs sampler proceeds as follows: Given a random vector $X = (x_0, \ldots, x_d)^T$ of length d and a target probability density function $\pi(X)$, in each iteration a component x_j, with $0 \leq j \leq d$ is picked. A proposal for the new state $\hat{x}_j$ of the component is generated according to

$$\hat{x}_j \sim \pi(x_j | x_0, \ldots, x_{j-1}, x_{j+1}, \ldots, x_d)$$

and accepted with probability equal to one. The same procedure is then repeated d-times to produce a single output of the algorithm.

Gibbs sampling is a special case of the Metropolis-Hastings algorithm with a proposal distribution that is given by the conditional distributions of the target pdf and an acceptance probability that is always equal to one.

To see this relation, let us denote by $\hat{X} = (x_o, \ldots, x_{j-1}, \hat{x}_j, x_{j+1}, \ldots, x_d)^T$ the state vector after an update by the Gibbs sampler. Then the pdf of generating $\hat{X}$ from X is given by

$$q(\hat{X}|X) = \pi(\hat{x}_j | x_0, \ldots, x_{j-1}, x_{j+1}, \ldots, x_d)$$

Conversely, the pdf of generating X from $\hat{X}$ is given by:

$$q(X|\hat{X}) = \pi(x_j | \hat{x}_0, \ldots, \hat{x}_{j-1}, \hat{x}_{j+1}, \ldots, \hat{x}_d)$$

Let us further denote the random vector that does not include the updated component as

$$X_{/j} = (x_o, \ldots, x_{j-1}, x_{j+1}, \ldots, x_d)^T$$

Note that X and $\hat{X}$ only differ by the component x_j, therefore $X_{/j} = \hat{X}_{/j}$.

By substituting into the update probability of the Metropolis-Hastings algorithm, we get

$$\alpha(X, \hat{X}) = \frac{\pi(\hat{X})}{\pi(X)} \frac{q(X|\hat{X})}{q(\hat{X}|X)} = \frac{\pi(\hat{X})}{\pi(X)} \frac{\pi(x_j|\hat{X}_{/j})}{\pi(\hat{x}_j|X_{/j})} = \frac{\pi(\hat{x}_j|\hat{X}_{/j})\pi(\hat{X}_{/j})}{\pi(x_j|X_{/j})\pi(X_{/j})} \frac{\pi(x_j|\hat{X}_{/j})}{\pi(\hat{x}_j|X_{/j})} = 1$$

Target pdfs that are in principle suited for the Gibbs sampler need to have a Markovian neighborhood structure, i.e. in the context of image processing, the pdf of a single pixel does only depend on pixels in a local neighborhood. This ensures that the conditional pdfs are relatively simple.

Note that the Gibbs sampler is only applicable for multivariate distributions (whereas the Metropolis-Hastings algorithm is also able to sample from univariate distributions).

Algorithm 3 summarizes the basic steps of the Gibbs sampler.

The index j is typically chosen at random from an uniform distribution (referred to as random scan). While this approach ensures reversibility of the chain, there are implementation scenarios (for example GPU-based implementations) that may benefit from a systematic scan, i.e. where the indices are chosen according to some predetermined order. While the systematic scan approach does not yield a reversible chain, convergence to the distribution $\pi(.)$ is still guaranteed.

Algorithm 3 Gibbs sampling

```
Choose an initial state $X_0 = \{x_0^0, \ldots, x_d^0\}$
Set $k = 0$
loop {Gibbs iteration}
    Choose an index $0 \leq j \leq d$
    Draw $x_j^{k+1} \sim \pi(x_j | x_o^k, \ldots, x_{j-1}^k, x_{j+1}^k, \ldots, x_d^k)$
    $k \Leftarrow k + 1$
end loop
```

4.2.2 Metropolis-within-Gibbs

With focus on the sampling of variational energies that incorporate regularizers that are based on spatial derivatives, it becomes clear that the Gibbs sampler is not directly applicable for most models, because of the non-standard form of the conditionals. It is clear however that those models do have a Markovian structure, and we can take advantage of the small spatial dependence of a single pixel to its neighboring pixels. We augment the scanning and update scheme of the Gibbs sampler with an additional acceptance/reject-step. The resulting algorithm is called Metropolis-within-Gibbs and forms the basis for sampling the variational energies covered in this thesis.

The algorithm is different from the classical Metropolis-Hastings algorithm in a simple detail. Assuming a multivariate target distribution, only a single component of the state vector is updated, similar to the Gibbs sampler. The update for the component is generated according to an arbitrary univariate proposal distribution and accepted or rejected according to the usual Metropolis-Hastings acceptance/rejectence probabilities.

Algorithm 4 formalizes the Metropolis-within-Gibbs sampler.

Algorithm 4 Metropolis-within-Gibbs

```
Choose an initial state $X_0 = (x_0, \ldots, x_d)$
Choose a proposal distribution $q$
Set $k = 0$
Choose an oversampling ratio $R > 0$
loop {Metropolis-Hastings iteration}
    for $n = 1 \ldots R$ do
        Choose an index $0 \leq j \leq d$
        Draw $y \sim q(.|x_j)$
        $X_{k+1/2} \Leftarrow (x_0, \ldots, x_{j-1}, y, x_{j+1}, \ldots, x_d)$
        $\alpha \Leftarrow \frac{\pi(X_{k+1/2}) q(X_k | X_{k+1/2})}{\pi(X_k) q(X_{k+1/2} | X_k)}$
        $X_{k+1} \Leftarrow X_{k+1/2}$ with probability $\min(\alpha, 1)$
        $X_{k+1} \Leftarrow X_k$ with probability $1 - \min(\alpha, 1)$
    end for
end loop
```

This modification, although small, heavily impacts the applicability of the algorithm for image processing applications. Especially in the context of models that are based on spatial derivatives (thus a single pixel is connected only with pixels in a relatively small neighborhood), this formulation offers tremendous benefits. In the standard version of the Metropolis-Hastings algorithm, each iteration needs a proposal of full dimension (i.e. a random image the same size as the input image has to be generated). It is relatively unlikely that such an update increases the probability of the pdf, if the proposal distribution is not perfectly matched to the shape of target distribution, resulting in slow movement of the chain.

The Metropolis-within-Gibbs updates only a single pixel, which by itself has a relatively large probability of moving the distribution to higher values (even with proposals as simple as a uniform distribution). The price for this simplicity is a very large correlation between two successive samples, which, however, can be mitigated by taking only every R-th image as output of the algorithm. This is called oversampling and reflected by the constant R in Algorithm 4. While the oversampling ratio R can in principle arbitrarily be chosen, we fix it to $R = |\Omega|$ for the remainder of this thesis.

The second advantage is given by the fact that if the target distribution is Markovian, large parts of $\pi(.)$ can be factored out, which allows the computation of the update probability $\alpha(x, y)$ for a single step by evaluation $\pi(.)$ only in a small neighborhood. This strongly alleviates the evaluation of the update probabilities. Moreover, such a scheme naturally lends itself to parallelization as all pixels that are conditionally independent can be updated at once (which of course is only practicable if the order in which components are updated is chosen according to a systematic scan).

4.2.3 Metropolis-Adjusted Langevin Algorithm

All of the presented algorithms so far are agnostic to the local characteristics of the target distribution. The movement of the Markov chain is dominated by the proposal distribution. It is possible, however, to speed up the movement of the chain to regions of high probability by exploiting the structure of the target distribution.

In [Grenander and Miller, 1994], the idea of using a Langevin diffusion to steer the sampling process was introduced. Langevin diffusions are a class of stochastic differential equations that come from physics and were originally used to describe Brownian motion, i.e. the movement of particles in fluids due to thermal noise.

The Langevin diffusion equation is given by

$$dX_t = dB_t + \frac{1}{2}\nabla \log(\pi(X))dt \tag{4.10}$$

where $t \in \mathbb{R}$ denotes the time, $B_t \in R^{|MN|}$ is a Wiener process and X_t denotes the current state of the , now continuous in time, random process that is described by the equation.

Note that (4.10) describes a stochastic process due to the stochastic nature of the Wiener process B_t. To simulate this process, it is necessary to discretize (4.10). An Euler discretization [Roberts and Tweedie, 1996] with a time-step increment of Δt leads to

$$\frac{X_{n+1} - X_n}{\Delta t} = B_{n+1} - B_n + \frac{1}{2}\nabla \log(\pi(X_n))$$

Rearranging with respect to X_{n+1} leads to

$$X_{n+1} = X_n + \frac{\Delta t}{2}\nabla \log(\pi(X_n) + \Delta t(B_{n+1} - B_n) \tag{4.11}$$

The difference $B_{n+1} - B_n$ of two successive realizations of a Wiener process is by definition normally distributed with zero mean and unit variance, i.e

$$B_{n+1} - B_n \sim \mathcal{N}(0, I)$$

This finally leads to the conclusion that the discrete-time approximation to the Langevin diffusion can be simulated by a Normal distribution, according to

$$X_{n+1} \sim \mathcal{N}(X_n + \frac{\Delta t}{2}\nabla \log(\pi(X_n)), (\Delta t)^2 I) \tag{4.12}$$

The discrete Langevin diffusion (which is often referred to as “Unadjusted Langevin Algorithm” (ULA) in the MCMC literature) obviously describes a Markov chain. Moreover, under certain circumstances, a simulation of 4.12 will produce a Markov chain with invariant distribution $\pi(X)$.

Roberts and Tweedie [1996] show conditions on when the continuous-time Langevin diffusion converges to $\pi(X)$: Apart from the obvious requirement that $\log(\pi(X))$ has to be continuously differentiable, the additional constraint

$$\nabla \log \pi(X) X \leq a|X|^2 + b, \qquad |X| > N$$

for some $N, a, b < \infty$ ensures convergence to the target distribution π.

Those results cannot simply be extended to the discrete approximation (4.12). The diffusion may or may not converge to π, depending on the actual properties of π, as the discretization perturbs the behavior of the diffusion process.

To still allow convergence to π, it is needed to add an additional Metropolis-Hastings accept-reject step to the algorithm [Roberts and Tweedie, 1996], which finally results in the Metropolis-adjusted Langevin Algorithm (MALA). Thus, in MALA, first a candidate update step is drawn according to the ULA update:

$$X_{n+1/2} \sim q(X_{n+1/2}|X_n) = \mathcal{N}(X_n + \frac{\Delta t}{2} \nabla \log \pi(X_n), \Delta t^2 I)$$

and accepted with probability

$$\alpha(X_n, X_{n+1/2}) = \min \left\{ \frac{\pi(X_{n+1/2}) q(X_n|X_{n+1/2})}{\pi(X_n) q(X_{n+1/2}|X_n)}, 1 \right\}$$

or rejected with probability

$$1 - \alpha(X_n, X_{n+1/2})$$

Due to the statistical independence of the individual components of the ULA proposal, this idea can also be applied to Gibbs fields. By simply setting the proposal distribution in Algorithm 4 to the ULA proposal distribution (4.12), we get the MALA-within-Gibbs algorithm.

4.3 Optimal Scaling

The scaling of the proposal distribution is a crucial factor in convergence speed. If the scaling is too small, the chain is not able to explore the space in reasonable time. If the scaling is too large, the chain will inevitably move into states that will be rejected too often. In both cases, the result is a poor convergence time of the sampling algorithm.

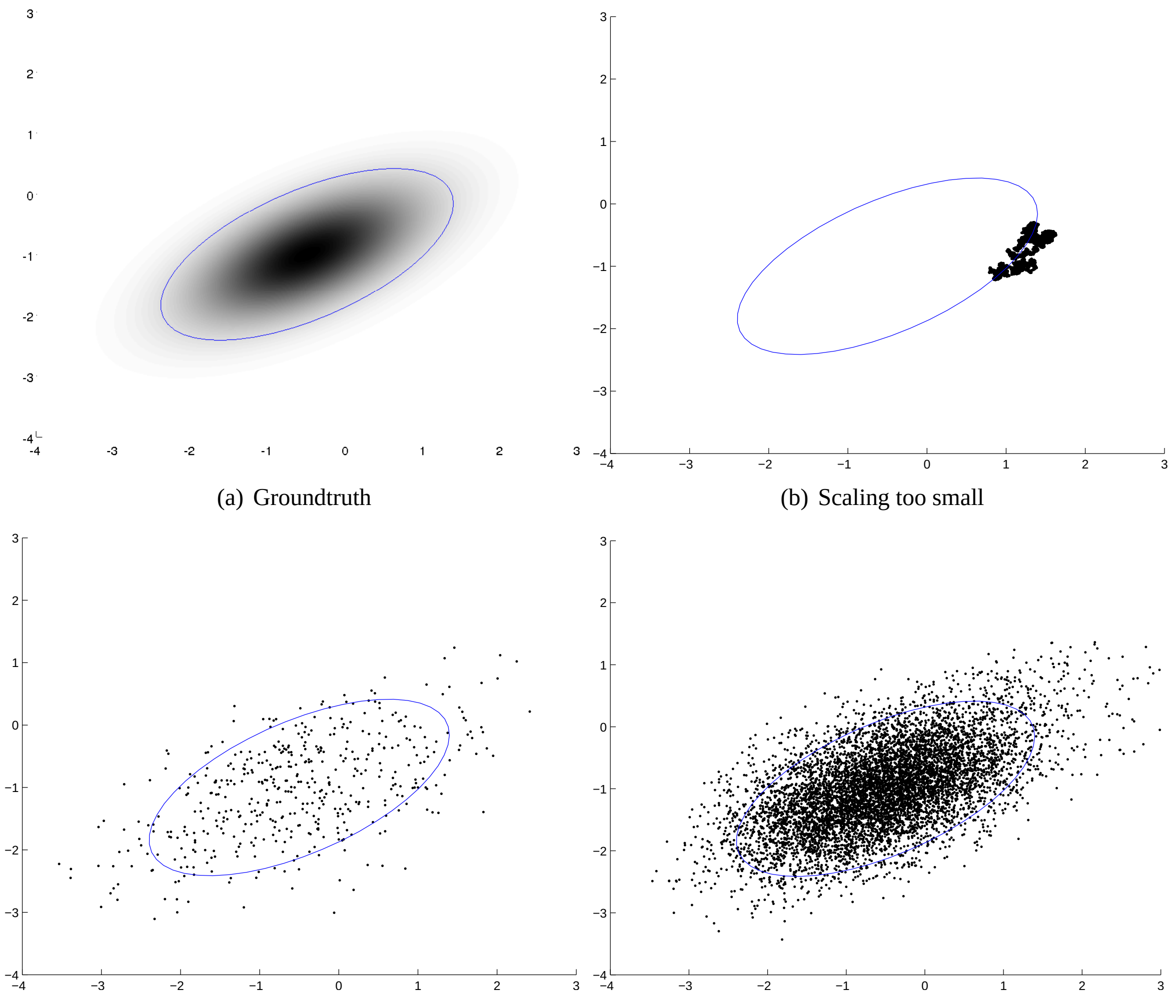
Figure 4.2 shows the impact of the scaling of the proposal distribution on the convergence time of the sampling algorithm (the example is due to [Louchet, 2008]). A bivariate normal distribution was sampled using the Metropolis-Hastings algorithm (as presented in Algorithm 2) where the proposals were generated according to a uniform distribution, with scaling α:

$$X_{k+1/2} \sim X_k + \alpha U_{[-1,1]}$$

Figures 4.2(b)-(d) show the first 10000 samples of the resulting chain for different values of α. If α is too small, most of the proposals are accepted, but the chain moves too slowly in space to explore the target distribution in reasonable time (Figure 4.2(b)). Figure 4.2(c) shows the converse case: The proposals are too large, hence most of the proposals are discarded. Only if the scaling is reasonably chosen (either by an automatic procedure or by hand), a good approximation of the target distribution can be obtained (Figure 4.2(d)).

Neal and Roberts analyze the convergence rates of Metropolis-Within-Gibbs and MALA-Within-Gibbs in [Neal and Roberts, 2006]. Their analysis is based on the acceptance rate τ_{accept}

$$\tau_{accept} = \frac{\text{\# of accepted proposals}}{\text{Total \# of proposals}}$$



(a) Groundtruth

(b) Scaling too small

(c) Scaling too large

(d) Optimal scaling

Figure 4.2: Sampling of a bivariate normal distribution (mean = $(-0.5, -1)$, Cov = $(0.9, 0.4; 0.4, 0.5)$ with different proposal scalings. (a) Groundtruth. The area of the ellipse covers approximately 86% of the total probability mass. (b) $\alpha = 0.01$: The proposals are too small, almost every proposal is accepted ($\approx 99\%$). (c) $\alpha = 6$: The proposals are too large, only a small fraction of the proposals is accepted ($\approx 4\%$). (d) $\alpha = 0.6$: Nearly optimal scaling, approximately 70% of the proposals are accepted

of the algorithm.

They suggest that Metropolis-Within-Gibbs optimally converges if $\tau_{accept} \approx 0.23$. MALA-Within-Gibbs allows for a higher acceptance rate where roughly half of the generated proposals are actually accepted ($\tau_{accept} \approx 0.57$). Both results assume a Gaussian proposal distribution, but experiments show that these results approximately hold for a uniform proposal distribution as well.

For many MCMC applications, it is feasible to hand tune the parameter to match the optimal acceptance rate. This method is not applicable in the context of image processing, as it would need multiple runs and hand-tuning of a parameter for every single image. We instead try to adapt the optimal scaling as the chain evolves in time using a stochastic gradient descent algorithm.

The adaption procedure presented is largely motivated by [Atchadé, 2006] and [Haario et al., 2001] and proceeds as follows:

Let us denote the current scaling by h_k and the optimal acceptance rate as τ_{opt}. The proposal distribution is fixed to be Gaussian and for a single pixel is given by

$$X_{k+1/2} \sim \mathcal{N}(f(X_k), h_k)$$

where $f(x) = x$ for Metropolis-within-Gibbs and $f(x) = x + \frac{\Delta t}{2}\nabla \log \pi(x)$ for MALA-within-Gibbs. Let us further introduce the expected acceptance rate $\tau(h)$ under the target distribution π for a given scale:

$$\tau(h) = \mathbb{E}_\pi \left\{ \int_S \alpha_h(x,y) q_h(y|x) dy \right\} = \int_S \pi(x) \int_S \alpha_h(x,y) q_h(y|x) dy dx$$

The difference of $\tau(h)$ from its optimal value can be measured by $(\tau_{opt} - \tau(h))^2$. This difference can be minimized by a simple steepest gradient scheme, as it was presented in chapter 2, which finally leads to an iterative update procedure for h:

$$h_{k+1} = h_k + \gamma_k(\tau(h_k) - \tau_{opt}) \tag{4.13}$$

where $\gamma_k > 0$ controls the step-size of the update.

In most scenarios, it is clearly not possible to compute $\tau(h)$. We can, however, estimate this function for a given h:

Consider that the chain is in state X_k and an oversampling ratio R is used. The scale h of the proposal distribution is changed only after R iterations. The transition kernel is therefore again constant inside the oversampling iteration, and we can approximate $\tau(h)$ by

$$\tau(h) \approx \frac{1}{R} \sum_{i=1}^{R} \alpha(X_{k+i-1}, X_{k+i}) q(X_{k+i}|X_{k+i-1}) \approx \frac{a}{R}$$

where a denotes the number of accepted proposals in the iteration. Using this estimate, h_k is updated after each oversampling iteration.

Note that (4.13) in principle is a stochastic procedure as τ_h is a random variable. Convergence of h to the optimal value is obvious but the procedure violates a fundamental assumption that had been made so far: The time-invariance of the transition kernel of the overall chain. After each oversampling iteration, the scaling of the proposal distribution, and therefore the transition kernel is modified. Hence neither convergence of the chain to the target distribution π nor ergodicity is guaranteed.

Roberts and Rosenthal [2007] show however that MCMC algorithms with a quite general class of time-varying transition kernels can remain π-irreducible and ergodic, provided some rather lax conditions are met, which, to cite [Roberts and Rosenthal, 2009], “provide a hunting license to look for adaptive MCMC algorithms”:

Let $\{P_\gamma\},\ \gamma \in \mathcal{Y}$ denote the set of transition kernels that are produced by the adaption procedure (4.13). Let further P_{Γ_n} denote the transition kernel at time-step n where Γ_n is a random variable itself (representing the uncertainty in the update procedure). Then the adaptive MCMC algorithm is ergodic with stationary distribution π if the following conditions are met:

Condition 1. *([Roberts and Rosenthal, 2007])*
Let

$$||f_1(x,.) - f_2(x,.)||_{TV} = \sup_{A \in \mathcal{B}} |f_1(x,A) - f_2(x,A)|$$

denote the Total Variation distance between two probability measures f_1 and f_2.

Then an adaptive MCMC algorithm is ergodic, if

a) (Diminishing Adaption:) The amount of adaption vanishes, as n goes to infinity:

$$\lim_{n\to\infty} \sup_{x \in X} ||P_{\Gamma_{n+1}}(x,.) - P_{\Gamma_n}(x,.)||_{TV} = 0$$

b) (Simultaneous Uniform Ergodicity:) For all $\epsilon > 0$, there exists a time $N = N(\epsilon)$ such that for all $x \in S$ and $\gamma \in \mathcal{Y}$

$$||P_{\gamma}^{N}(x,.) - \pi(.)||_{TV} \leq \epsilon$$

The condition of diminishing adaption can be enforced in our adaption procedure by gradually reducing the step size, for example by setting $\gamma_k = \frac{C}{k}$.

The second condition basically states that every kernel has to be ergodic with stationary distribution π, i.e. every kernel for itself represents a valid MCMC procedure. Moreover, all kernels have to exhibit the same convergence rate. Now note that in our adaption scheme merely the scaling of the proposal distribution is modified. Using the assumption that the proposal distribution is a Gaussian and that the target distribution is log-concave, this condition is always met.

4.4 Chapter Summary

Based on the need to solve a very high-dimensional integral, this chapter introduced some fundamental aspects of Markov Chain Theory and several algorithms to generate Markov chains, which effectively sample from a desired target distribution. The presented algorithms are both simple and general (in the sense that only weak or no assumptions on the target distribution are made) and therefore perfectly suited for the application to variational models.

LSE estimation can be carried out via the ergodicity theorem (1), which essentially states that the expected value of a target distribution can be approximated by averaging the output of a sampling algorithm. While this approximation is very simple, the sampling algorithms themselves are usually computationally very demanding for high-dimensional problems. Moreover, there are some parameters that have to be tuned on a case-by-case basis, which lessens the real-world applicability of those algorithms. The following chapters are therefore concerned with the refinement of the proposed sampling algorithms to allow both optimal convergence speed and massive parallelization.

Chapter 5

From Samplers to Estimators

Now that we are able to construct a Markov Chain that samples from a target distribution, we concentrate our attention again on our initial goal, the approximation of high dimensional integrals, to estimate the expected value of a variational model. Recall again that the integral (i.e. the LSE estimator) that is to be approximated is given by:

$$u^* = \mathbb{E}\{u\} = \int_{\mathbb{R}^{|\Omega|}} up(u|f)du$$

And via Theorem 1, we can approximate this integral by

$$u^* \approx \frac{1}{N}\sum_{k=1}^{N} U_k \tag{5.1}$$

provided that the samples U_k are distributed according to $p(u|f)$. So, to approximate the LSE estimator, we can therefore run any feasible algorithm presented in the preceding chapter and simply compute the arithmetic mean of the generated images. However, for an efficient implementation, two additional points have to be considered:

- After starting the evolution of the chain, the samples are not immediately distributed according to the target distribution. Therefore, the chain has to be run for some time before samples should be incorporated into the arithmetic mean.
- A stopping criterion to assess convergence of (5.1) is needed.

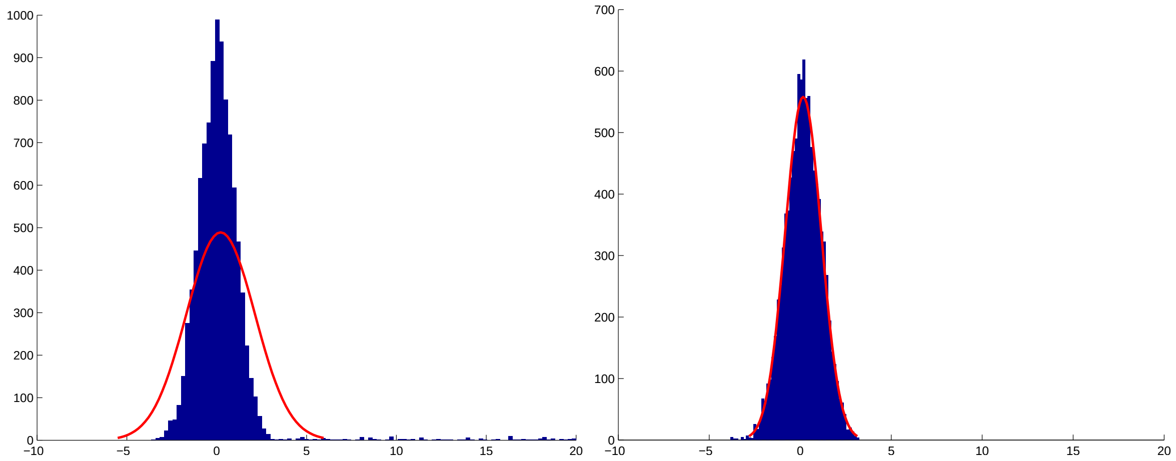
Both problems cannot be tracked analytically and strongly depend on the target distribution. An algorithm that heuristically tackles both problems simultaneously will be presented in section 5.1.

The sampling algorithms presented in the preceding chapter open up another possibility: Via the sampling procedure, we gain important insight into the target distribution, which can be used to find the MAP estimate. Section 5.2 introduces a popular algorithm, called “Simulated Annealing”, which exhibits asymptotic convergence to the global optimum, even in non-convex energies.

5.1 Burn-in and Convergence Control

After starting the evolution of the chain, samples are not immediately distributed according to $\pi(.)$, except for the case where the initial image U_0 is drawn from $\pi(.)$ itself.

The time needed to reach equilibrium is called the burn-in time. When estimating moments, one relies on a set of samples from the target distribution. To obtain a reliable estimate, it is therefore crucial that samples that were not sampled from the invariant distribution are not incorporated into the estimate as they would potentially distort the LSE estimate for a very long time.



(a) Without burn-in

(b) With burn-in

Figure 5.1: Comparison of the sampling of a standard normal distribution with and without burn-in. The examples show the distribution of 10000 samples with a fitted normal distribution superimposed (red). In both examples, the chain was initialized at a state with very low probability (x = 20). (a) Without burn-in. The samples are not immediately distributed according to a standard normal distribution, because the chain needs some time to reach regions of higher probability. (b) With a burn-in (2000 samples). The resulting distribution of the samples fits the target distribution.

Figure 5.1 illustrates this effect. A standard normal distribution $\mathcal{N}(0, 1)$ was sampled using a starting value with low probability ($x = 20$). The chain needs some time to reach regions with larger probability, which effectively distorts the LSE estimate (Figure 5.1(a)). If the first samples where the chain moves towards regions of higher probability are discarded, the resulting distribution is much closer to the target distribution (Figure 5.1(b)).

An estimator therefore needs to throw away the first b samples. Unfortunately it is not possible to determine b analytically or by just observing a single run of the chain. There are many approaches that try to heuristically determine good values for b. One such approach is to run two or more chains (that were initialized at different starting values) in parallel and compute the arithmetic means for different burn-in values. As the chains reach equilibrium and the burn-in values increase, the averages will eventually converge to the same value, indicating both an optimal burn-in value as well as convergence of the LSE estimate. Such a scheme, however, demands to either store every sample or to do a preliminary run to determine the burn-in value before the actual LSE estimation is carried out. The first option is infeasible in the context of image processing, and the second option is quite inefficient.

Louchet [2008] developed an algorithm that addresses these problems, which is reproduced in Algorithm 5. While the algorithm is largely based on heuristics, it actually performs well in practice.

Let us briefly explain the idea behind the algorithm:

The algorithm runs two chains, $\{U_n\}$ and $\{\hat{U}_n\}$, that are generated using identical proposal distributions and different starting values. The Metropolis-Hastings iteration (MH iteration) refers to a full scan of length $|\Omega|$, systematic or random, of either the Metropolis-within-Gibbs or the MALA-within-Gibbs sampler. The acceptance rate of the first chain is used to infer the optimal scaling for both chains.

By taking into account the burn-in time b, the LSE estimators for both chains are then given by

$$S_n^b = \frac{1}{n-b} \sum_{k=b+1}^{n} U_k \qquad \hat{S}_n^b = \frac{1}{n-b} \sum_{k=b+1}^{n} \hat{U}_k$$

First, note that by averaging both sums

$$G_n^b = \frac{S_n^b + \hat{S}_n^b}{2} \tag{5.2}$$

we obviously can get a better estimate than by considering a single sum alone because this is roughly equivalent to running a single chain twice as long.

[Louchet, 2008] states, largely based on empirical results, that the distance of this refined estimate to the true value can be approximated by the distance between the estimates of the individual chains:

$$\|G_n^b - u^*\| \approx \frac{1}{2}\|S_n^b - \hat{S}_n^b\| \tag{5.3}$$

This leads to the stopping criterion:

$$\|S_n^b - \hat{S}_n^b\| \leq 2\epsilon$$

which ensures that the output (5.2) is roughly a distance of ϵ away from the true value u^*. The considerations so far assumed a known burn-in time b, which could be computed for a fixed run-time n if u^* was known. Using (5.3) again, the optimal burn-in time can be approximated by:

$$\hat{b} = \arg\min_{b\in\{1,\ldots,n\}} \|G_n^b - u^*\| \approx \arg\min_{b\in\{1,\ldots,n\}} \|S_n^b - \hat{S}_n^b\| \tag{5.4}$$

Now note the minimum in (5.4) involves the full length n of the chains. To find this minimum, this would require to store each image generated by the sampling algorithm (or equivalently to store every partial sum). This is of course impossible, considering that typically hundreds to thousands of images are generated until the algorithm converges.

It is not necessary, however, to consider every index from $1 \ldots n$ as a potential burn-in candidate. If one considers only a subset of those indices, the worst result would be a few wasted iterations, where the algorithm keeps running even if the stopping criterion was already met.

Algorithm 5 LSE Estimation ([Louchet, 2008])

```
Set $n = 0$, $\lambda = 1.2$, $\alpha_0 = 1$
Generate $U_0$ and $\hat{U}_0$, with uniformly i.i.d. pixels
Set $S_0 = 0$, $\hat{S}_0 = 0$
repeat
    Compute $U_{n+1}$ and $\tau_{n+1}$
    $\{$MH iteration with proposal distribution $\mathcal{N}(f(U_n), \alpha_n)\}$
    Compute $\hat{U}_{n+1}$
    $\{$MH iteration with proposal distribution $\mathcal{N}(f(\hat{U}_n), \alpha_n)\}$
    $S_{n+1} \leftarrow S_n + U_{n+1}$
    $\hat{S}_{n+1} \leftarrow \hat{S}_n + \hat{U}_{n+1}$
    if $n \in \lfloor\lambda^{\mathbb{N}}\rfloor$ then
        Store $S_{n+1}$ and $\hat{S}_{n+1}$
        Erase $S_k$ and $\hat{S}_k$ with $k < n/6$
        Set $\hat{b} = \arg\min_{b\in\lfloor\lambda^{\mathbb{N}}\rfloor} \|\frac{S_n - S_{\hat{b}}}{n-\hat{b}} - \frac{\hat{S}_n - \hat{S}_{\hat{b}}}{n-\hat{b}}\|$
    end if
    $\alpha_{n+1} \leftarrow \alpha_n + \frac{1}{n+1}(\tau_{n+1} - \tau_{opt})$
    $n \leftarrow n + 1$
until $\|\frac{S_n - S_{\hat{b}}}{n-\hat{b}} - \frac{\hat{S}_n - \hat{S}_{\hat{b}}}{n-\hat{b}}\| \leq 2\epsilon$
return $\frac{1}{2}\frac{S_n - S_{\hat{b}}}{n-\hat{b}} + \frac{1}{2}\frac{\hat{S}_n - \hat{S}_{\hat{b}}}{n-\hat{b}}$
```

This idea is reflected in the algorithm by a geometric grid: The algorithm considers only $b \in \lfloor 1.2^{\mathbb{N}}\rfloor = \{\lfloor 1.2^k\rfloor, k \in \mathbb{N}\}$ as candidate for the burn-in parameter. Moreover, there is no need to actually store each image in this sequence. It suffices to examine only the images, where $b \geq n/6$ (again from empirical arguments in [Louchet, 2008]). This results in at most 10 images that are stored per chain at any time, totaling 20 images, which is tractable on most computers.

The initializations U_0 and $\hat{U}_0$ can in principle be arbitrarily generated as long as they are not too similar, which would break the stopping criterion. Harris-recurrence and π-irreducibility guarantee that the chain forgets about its initial state in finite time. Experiments show that the chains approach each other very rapidly (after 10-100 iterations) regardless of the specific initialization. The impact of the initialization on the convergence speed is therefore negligible. For our specific implementation, we used a random initialization with uniform, i.i.d. pixels.

5.2 Simulated Annealing

It has already been mentioned that a sampling algorithm could in principle also be used to infer the MAP estimate. A naive approach would simply store the image with highest probability along the evolution. If the sampling algorithm is run long enough, it is guaranteed that the image with highest probability in the run is sufficiently near to the true MAP estimate. Such an approach would, however, need very long runs for non-convex optimization problems (due to the tendency of the sampling algorithms to get stuck in local optima for a long time) and would be of little value for convex optimization problems (due to the availability of much faster algorithms).

Simulated Annealing [Kirkpatrick et al., 1983], a modification of the Metropolis-Hastings algorithm, is able to considerably speed-up global MAP inference for non-convex energies, compared to the naive sampling approach.

In order to minimize an energy $E(u)$, one can alternatively sample from the distribution

$$p(u) = \frac{1}{Z} \exp - \frac{E(u)}{T} \tag{5.5}$$

where T is gradually reduced. The maximum of the pdf then corresponds to the minimum of the energy $E(u)$. The parameter T is called the temperature due to the resemblance of (5.5) to the Boltzmann distribution from thermodynamics where it denotes the temperature of a gas.

Recall again how the Metropolis-Hastings algorithm moves within a target distribution:

- If the proposed state has higher energy than the previous state, the state is always accepted. Such a move effectively brings the chain nearer to a, potentially local, maximum.
- Conversely, if the proposed state has lower energy than the previous state, the proposal is accepted with non-zero probability. This probability is inversely proportional to the energy difference between the states. The algorithm can therefore move “downhill” in the pdf, which effectively allows to escape local maxima.

Figure 5.2 depicts the influence of the temperature on the shape of a simple distribution. For high temperatures, the resulting distribution becomes more uniform, resulting in smaller energy differences between jumps. This allows the Metropolis-Hastings algorithm to rapidly move around in the target distribution and avoids being trapped in local optima. As the temperature decreases, the maxima of the function become more and more peaked. Provided that the annealing schedule (i.e. the rate at which the temperature is decreased) is slow enough, the sampler will become trapped in the maximal mode. As the temperature T approaches zero, “downhill” movements become very unlikely, eventually leading the sampler to the global maximum of the target distribution.

Algorithm 6 shows the full algorithm, with automatic tuning of the scaling of the proposal distribution. The algorithm has several variables which have to be determined based on the target distribution. The cooling schedule is fixed to a logarithmic schedule, where γ is nearly (but not equal to) one.

Algorithm 6 Simulated Annealing

```
Set $n = 0$,
Choose $T_0 > 0$, $\alpha_0 > 0$, $K > 0$, $0 < \gamma < 1$
Generate $U_0$
repeat
    $k = 0$
    while $k < K$ do
        Compute $U_{k+1}$ and $\tau_{k+1}$
        $\{$MH iteration with target $\exp\{-E(u)/T\}$ and proposal $\sim \mathcal{N}(f(U_n), \alpha_k)\}$
        $\alpha_{k+1} \leftarrow \alpha_k + \frac{1}{n+1}(\tau_{k+1} - \tau_{opt})$
        $k \leftarrow k + 1$
    end while
    $\alpha_0 \leftarrow \alpha_k$
    $T_{n+1} = \gamma T_n$
    $n \leftarrow n + 1$
until $T_{n+1} < T_{min}$
return $U_{n-1}$
```

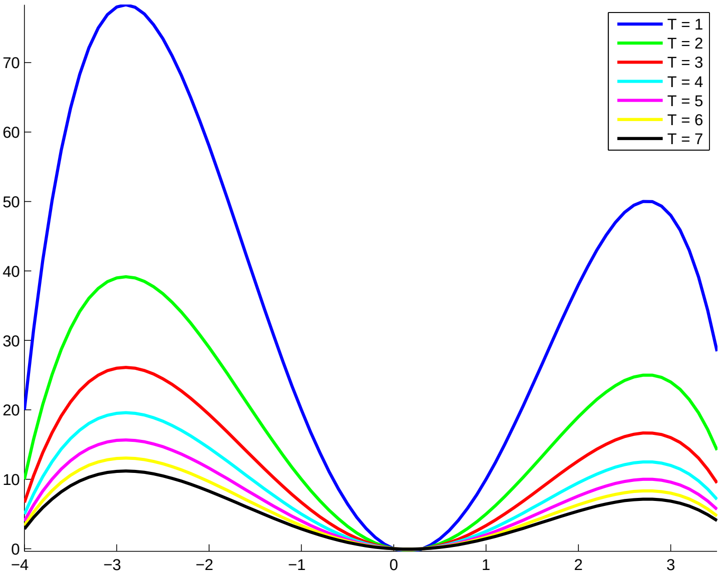


Figure 5.2: Logarithmic plot of the probability density induced by a quartic function ($x^4 - 16x^2 + 5x$) for different temperatures. For high temperatures, the distribution is nearly a uniform distribution. For smaller temperatures, the maxima become more and more pronounced.

Bélisle [1992] gives conditions for the convergence of the Simulated Annealing algorithm in a continuous setting:

Let $B_\epsilon = \{u \in S : E(u) \leq E(u^*) + \epsilon\}$ be the ball with radius ϵ centered at the global optimum u^*. Then the Simulated Annealing sequence u_n converges in probability to the global optimum, i.e.

$$\lim_{n \to \infty} Pr(u_n \in B_\epsilon) = 1, \qquad \forall \epsilon > 0$$

if the following conditions are met:

1. The time-variant Markov Chain has a transition kernel

$$R(x, A) = \int_A r(x, y) dy$$

where $r(x, y)$ satisfies $\inf_{x,y \in S} r(x, y) > 0$.

2. The set B_ϵ has positive Lebesgue measure.
3. For every open subset $G \in S$, $R(x, G)$ is continuous with respect to x.
4. For any U_0 and T_0, the sequence T_n converges to zero with a probability equal to one.

The first condition states that every state can be reached with non-zero probability in a single step. This means that the inner loop of Algorithm 6 has to be run long enough, or alternatively that the proposal distribution has to assign a non-zero probability to every possible state. The second condition states that the ball around the optimum has to contain a probability mass with respect to the posterior distribution. The third condition limits the influence of a small perturbation around the current state. The last condition is surprisingly lax as it states that the annealing schedule only has to converge in probability towards zero. This allows also adaptive annealing schedules, which however will not be examined here.

From a practical point of view, the design of a robust Simulated Annealing schedule involves some trial-and-error to find good parameters that ensure convergence (convergence in probability is a rather weak result). Parameters for specific applications can be found in chapter 7.

In our implementation, we used the Metropolis-Within-Gibbs sampler with a Gaussian proposal distribution. MALA can in principle be used, but has unfavorable convergence behavior, due to its strong attraction to local optima.

5.3 Chapter Summary

In this chapter, the concept of LSE estimation based on MCMC methods was refined to a fully automatic algorithm. We will use this algorithm exclusively for convex energies, however, because LSE estimates seldom provide meaningful results in non-convex high-dimensional posteriors (see chapter 3).

For non-convex energies, a global optimization algorithm, based on the Metropolis-Hastings algorithm, was presented. While only weak convergence results are available, the algorithm is useful nonetheless if the parameters are carefully selected for a given problem.

Both algorithms are computationally very demanding. A single sweep (iteration) of the Metropolis-Within-Gibbs or the MALA-Within-Gibbs samplers depends quadratically on the size of the input image. Typically, one needs hundreds to thousands of sweeps, resulting in very long run-times of the algorithms if they are implemented as sequential procedures (i.e. each pixel is visited one at a time). The following chapter is therefore concerned with a parallel implementation of the algorithms, which is able to shorten the run-time from several minutes to a few seconds.

Chapter 6

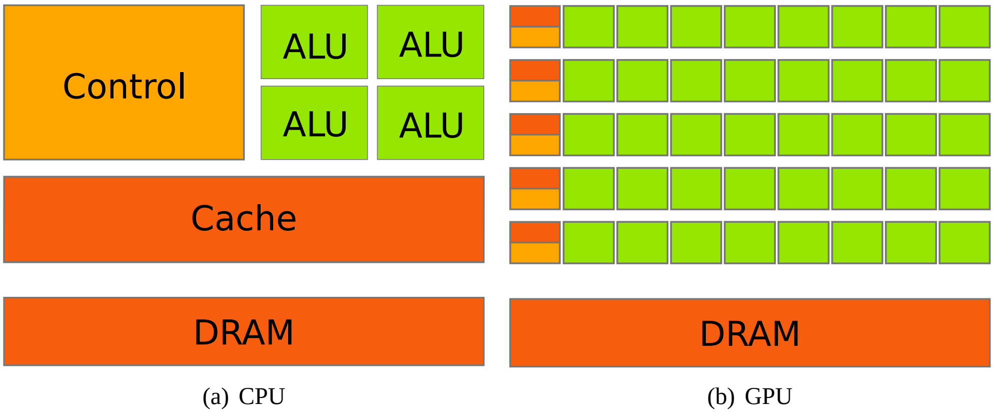
GPU Implementation

The sampling algorithms presented so far are computationally very demanding. Given a moderately sized image, even the simplest model, denoising using the ROF model, would take several minutes to produce a result when executed on a modern consumer-level CPU.

Note, however, that the sampling algorithms do not need full knowledge of the entire image to compute a single-pixel update probability. Only the neighborhood that is directly influenced by the pixel has to be considered. Given the Total Variation prior, approximated with finite forward differences, it therefore suffices to only look at two immediately neighboring pixels. Together with a systematic scan, this opens the possibility for massive parallelization.

The widespread availability of cheap, high-performance graphics cards, which are in fact massively parallel multiprocessing units, has recently spawned a trend in scientific computing: Instead of using expensive, dedicated multiprocessor computers, one relies on the Graphics Processing Unit (GPU) that is present in most desktop computers to perform general purpose computations (typically referred to as "GPGPU": General-Purpose computation on Graphics Processing Units).

GPUs are designed for the requirements of real-time 3D applications where large sets of data (i.e. vertices and pixels) have to be processed. In such applications, each vertex or pixel is typically processed by the same program. The architecture of GPUs reflects these requirements. They are designed for massive data-parallelism with less emphasis on flow control and memory caching, compared to typical CPU architectures (see Figure 6.1).



(a) CPU (b) GPU

Figure 6.1: Architecture of CPUs vs. architecture of GPUs [Nvidia, 2009]. In CPUs, a large portion of the total transistors are dedicated to flow control and memory caching. GPUs put less emphasis on flow control and caching but provide a large amount of ALUs for massively data-parallel computations.

To assist a programmer in the creation of GPGPU applications, three main frameworks have emerged:

- Nvidia CUDA[1] (Compute Unified Device Architecture): A framework developed by Nvidia specifically for their hardware platform.
- ATI FireStream[2]: The pendant to CUDA for AMD/ATI graphics hardware. Now superseded by OpenCL.
- OpenCL[3]: An open, standardized GPGPU framework. Implementations are available for AMD/ATI, Nvidia, as well as other, less common, hardware platforms.

Nvidia CUDA was the de facto standard for GPGPU computing in the last years and has a very mature and stable implementation. We therefore chose the CUDA framework (Version 2.3) for our implementation.

This chapter is organized as follows: Section 6.1 gives an overview of the CUDA framework. Section 6.2 describes some GPGPU-specific implementation details for the Metropolis-Within-Gibbs and MALA-Within-Gibbs samplers and shows a speed comparison to a CPU-bound implementation. Section 6.3 finally concludes with a brief summary of this chapter.

6.1 Nvidia CUDA

In the Nvidia CUDA framework, small programs (called kernels) that are subject to parallel execution can be coded using an extension to the C programming language. This allows to easily implement parallel algorithms to anyone who is familiar with C or similar languages. To achieve optimal performance, it is however necessary to consider the underlying programming model and device architecture as well:

When a kernel is called N threads execute the kernel in parallel. Those threads are, due to hardware constraints, organized into execution units, called blocks [Nvidia, 2009]. A number of blocks forms a grid, the largest execution unit (see Figure 6.2).

Each thread has a private memory space in the form of fast registers and slow local memory (which is mostly used for large automatic data structures that would consume too much register space). Threads in the same block can exchange data via shared memory, which is guaranteed to have low latency. Both memory types are volatile across kernel calls.

For data that needs to be persistent across kernel calls and has to be accessed by all threads, global memory can be used. This type offers read- and write-access for all threads but exhibits high latency. Access to global memory should therefore be kept at a minimum. Global memory can be bound to texture units, providing a fast, cached read-only memory. This type of memory is optimized for 2D spatial locality and offers special addressing modes for boundary handling and interpolation between neighboring texels, which makes it especially useful for image processing applications. Local and shared memory are not visible to the CPU. Data exchange between the CPU and GPU is possible via global memory and should again be kept at a minimum for optimal performance.

Both the block size (the number of threads per block) and the grid size (the number of blocks) can be chosen by the programmer. Each thread of a block is executed on the same multiprocessing unit of the GPU. This limits the number of possible threads in a block due to a limited amount of registers and shared memory on a core. Typical block sizes are 128, 256 or 512 threads per block and should be in general a multiple of two. The grid size is mainly governed by the size of the data that has to be processed. For imaging applications, typically one thread processes one pixels. The grid size therefore has to be chosen in a way that at least the whole image is covered with threads.

[1] http://www.nvidia.com/cuda

[2] http://developer.amd.com

[3] http://www.khronos.org/opencl

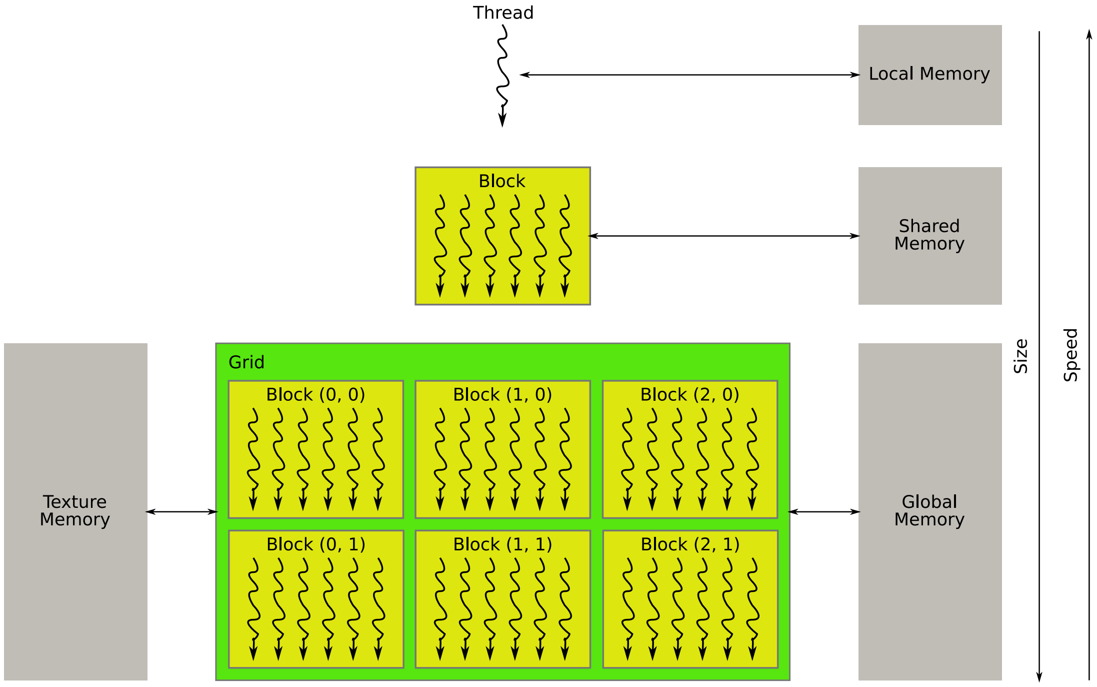


Figure 6.2: Memory hierarchy and thread grouping of the Nvidia CUDA platform [Nvidia, 2009].

6.2 Implementation Details

A striking characteristic of the Metropolis-Hastings algorithm is that no assumptions about the target distribution have to be made. We exploited this generality to build a “plug-and-play” system, which allows rapid implementation and testing of variational models.

Figure 6.3 shows a simplified UML diagram of the proposed system. It basically consists of a class that implements the sampling algorithm (`MH_Sampler`), which uses one or more `Terms`, each representing an additive term in the energy functional. Consider the ROF model as an example: To sample from this model, one registers an instance of `TV_Prior` and `L2_DataTerm` to the sampler. If desired, each term can additionally correct the proposal using an estimate of the gradient to make a MALA-Within-Gibbs step instead of a standard Metropolis step.

To perform a single oversampling iteration, the instance of `MH_Sampler` generates a proposal and passes the current state along with the proposal to the registered `Term` instances, which in turn generate a log-probability of moving to the proposed state. The sampler then decides if the proposal for each pixel is accepted or rejected according to the Metropolis-Hastings criterion and updates the state accordingly.

Finally, the classes `LSE_Estimator` and `MAP_Estimator` use the sampler to estimate statistics from the sampled distribution according to the algorithms presented in the preceding chapter.

This system is very flexible: One can implement arbitrary algorithms that rely on sampling, new terms can be added and one can form arbitrary combinations of terms to test different models.

Let us now identify parts of this system that are data-parallel, i.e. could benefit from an implementation using CUDA:

- Computation of the update probabilities
- Generation of the proposals and the Accept/Reject step in the Metropolis-Hastings sampler
- Mixing of chains for convergence control of the LSE estimator

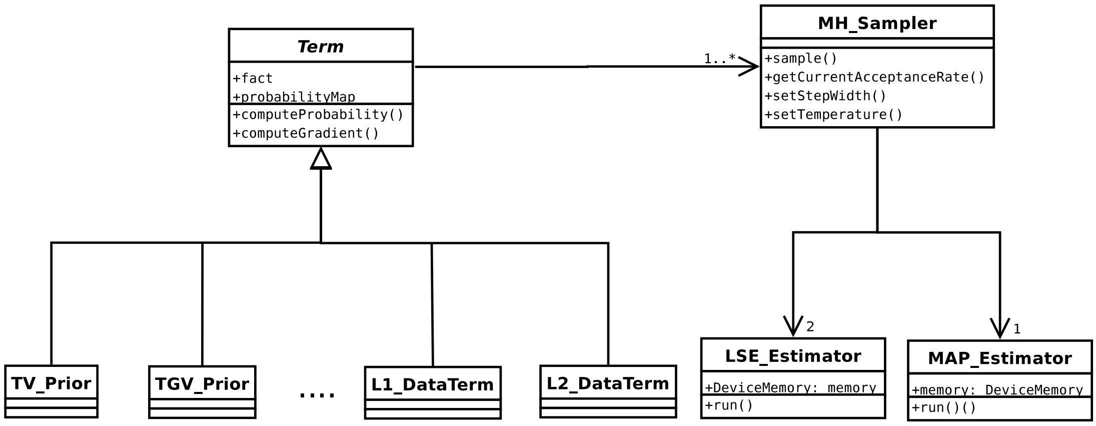


Figure 6.3: UML diagram of the proposed implementation

The mixing of the chains only requires standard linear algebra and reduction operations, a parallel implementation therefore is straightforward and will not be discussed here.

The generation of proposals and the corresponding update have to be subject to some considerations in a GPU-bound implementation. The first issue is the influence of the scan order (more precisely which pixels can be updated parallely). The second is the efficient generation of pseudo-random numbers on a GPU. We will review both issues in more detail in the following.

6.2.1 Influence of the scan order

The simplest parallel systematic scan order is given by a simultaneous update of all pixels. For each pixel a proposal is made and accepted based on the full conditional probability of the current pixel to all other pixels of the current state. Such a scheme is attractive due to its simplicity. Successive samples of such a scheme are however strongly correlated, which slows down the convergence of the sampling algorithm.

We propose a simple modification to lower the correlation between successive samples. Instead of updating all pixels at once, two sequential sweeps are made for a single iteration: Let i and j denote the coordinate of a pixel in the image. Then in the first sweep, all pixels where $i + j$ is even are updated. In the second sweep, all pixels with $i + j$ odd are updated, based on the conditionals of the previous sweep. Such an ordering is typically referred to as “Red-Black” iteration in the literature .

Figure 6.4 depicts the Red-Black updating scheme. The pixels that influence the probability of a single pixel update when using a TV prior with forward differences are hinted in gray in this image. Note that using a more sequential scanning strategy (i.e. updating only pixels that do not directly influence each others conditional probabilities) further reduces correlation between samples. This, however, hinders effective parallelization and does not increase performance in practice. Our experiments show that the red-black scan reduces the number of iterations till convergence is reached in algorithm 5 by roughly 30%.

6.2.2 Pseudo-Random Number Generators

Due to the lack of sources of true randomness on digital computers, applications that rely on randomness typically employ a Pseudo-Random Number Generator (PRNG). PRNGs are algorithms that deterministically, based on an initial seed value, generate a sequence of numbers that shares some statistical properties with true random numbers. In most algorithms the sequence is generated based on a recursive function, i.e. given a seed value x_0, the next value x_{n+1} can be computed as a function of x_n. Direct calculation of x_{n+1} is not possible in most algorithms (see [Blum et al., 1986] for an exception), rendering such a procedure inherently sequential and therefore problematic for a GPU-based implementation.

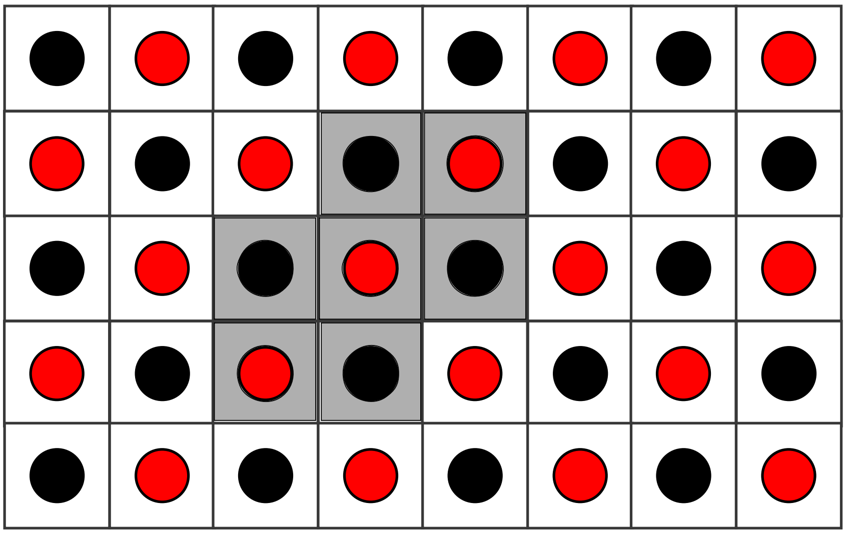


Figure 6.4: Red-Black updating scheme and single-pixel dependencies. In the first sweep, all red-pixels are concurrently updated. In the second sweep, the black pixels are concurrently updated. For a red update and a TV prior, the neighborhood dependencies for a single pixel are hinted in gray.

Another problem of a GPU-based PRNGs is the lack of high-precision integer arithmetic on GPUs. Typical PRNGs require computations with very large integers to allow a sufficiently long period of the generated sequence (i.e. the length of the sequence before it starts to repeat itself). This problem plays a larger role in cryptography applications than in Monte Carlo simulations. For simulations, we generally do not need pseudo-random numbers of cryptographic quality, which alleviates this problem and allows us to employ a simple and fast algorithm.

We chose an implementation that is based on the well-known `rand48()`-function from the C standard library. This function uses a Linear Congruential Generator (LCG) to generate a uniformly pseudo-random sequence. A fast GPU-based implementation that was specifically developed for simulation purposes is available at [van Meel and Amolf, 2010]. To generate Gaussian pseudo-random numbers from this sequence, we used the well-known polar transform (see [Thomas et al., 2007] for an overview of the generation of Gaussian random numbers).

Our experiments show that this implementation fits our needs, both in terms of speed and quality of the generated sequence. Note, however, that while the implementation is indeed fast, it nonetheless poses the biggest potential for optimization. Roughly 40% of the computation time of each Metropolis-Hastings iteration is concerned with the generation of random numbers. However, there seems to be no fast alternative for the parallel computation of pseudo-random numbers at the moment.

6.2.3 Speed comparison

A speed comparison shows the superiority of our GPU-based implementation to a sequential version. We compared the run-time of 1000 successive Metropolis-Hastings iterations for different image sizes. The sampled energy was the ROF model with parameters $\lambda = 0.4$ and $\sigma = 0.01$. All experiments were carried out on an Intel Xeon CPU with 2.53 Ghz and 24GB of RAM. The GPU code ran on a Nvidia Tesla S1070 computing system.

Table 6.1 summarizes the execution times. The parallel version roughly speeds up the computations by a factor of 35. Note that the LSE estimator seldom needs more than 2000 iterations for convergence and typically even converges after a few hundred iterations (although the convergence time is strongly dependent on the spread of the pdf that has to be sampled), making this algorithm indeed useful for real-world applications, when run on a GPU.

Size	GPU	CPU	Speedup
128x128	1.0	13.4	13.4
256x256	1.5	56.8	37.9
512x512	3.2	104.2	32.5

Table 6.1: Comparison of execution time in seconds between GPU and CPU implementation for different image sizes. The test consisted of 1000 full-scan Metropolis-Hastings iterations.

6.3 Chapter summary

In this chapter, the implementation of a Metropolis-Within-Gibbs algorithm for the sampling of variational models in image processing was outlined. Such algorithms are computationally very demanding and are therefore implemented to run on Graphics Processing Hardware, which allows massive parallelization of the proposed algorithms. We discussed problems and challenges that arise for GPU-based implementations of sampling algorithms and gave some hints on how those problems have been circumvented in our specific implementation. Finally, we showed a speed comparison of our GPU-based implementation to a CPU-based implementation. Our experiments show that the GPU version outperforms the CPU version by a factor of 35.

Chapter 7

Variational Models

In the following chapter, we will present experimental results of our approach to variational models.

The chapter is organized as follows: Section 7.1 applies the LSE estimator to two different denoising models and compares the results to the respective MAP estimates. Section 7.2 is concerned with the estimation of Optical Flow from images pairs. The examined models are non-convex and are usually solved via a convex approximation combined with MAP estimation. We will employ the Simulated Annealing procedure, introduced in chapter 5 to directly optimize those non-convex energies.

7.1 Denoising

In this section, the MAP estimator and the LSE estimator for different denoising models are compared. To quantitatively compare the denoising procedures, we chose the "Structural Similarity (SSIM) index" [Wang et al., 2004] as error metric. Unlike simple MSE-based error metrics, SSIM is able to capture visual differences between images better and is therefore more suitable for the comparison of reconstruction algorithms. The test images consisted of 13 images from the `DenoiseLab` database [Lansel, 2007], which were degenerated by different levels of noise. All MAP estimates in this section were obtained using the primal-dual algorithm that was presented in chapter 2.

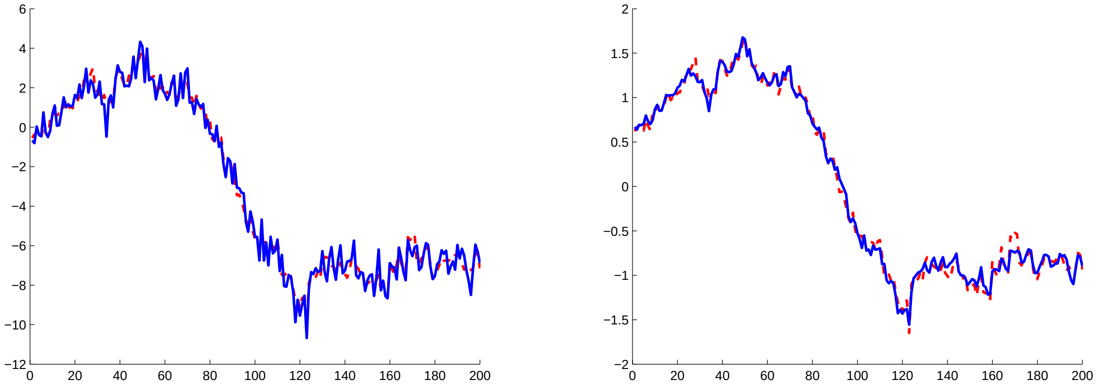
7.1.1 The ROF Model

In section 3.2, it was shown that solutions of the ROF models do not follow the probability distributions that were initially assumed in the construction of the model. Let us first empirically check the distributions that result from the LSE estimator of the ROF model.

Recall the random walk model introduced in section 3.2:

The Laplacian random walk is given by the sequence $x_{i+1} = x_i + L$, where L is distributed according to a zero-mean Laplacian distribution with variance β. The sequence x_{i+1} is then corrupted by zero-mean additive white Gaussian noise with variance σ^2. It was already mentioned that this sort of input data is in some sense the best-case input to the ROF model, as the data was generated according to the laws that form the basis of the denoising model. MAP estimates however, behave, poorly. Staircases emerge in the denoised function and the distribution of the result neither resembles a Laplacian distribution nor do the residuals resemble Gaussian noise (see Figures 3.1 and 3.2).

Figures 7.1 and 7.2 show the denoising of the Laplacian random walk using the LSE estimate of the ROF model and their respective distributions. The denoising result is far better, there are no staircases present in the reconstruction. Furthermore the distribution of the LSE estimate exhibits much more similarity to the true distributions. The distribution of the differences $x_{i+1} - x_i$ perfectly follows a Laplacian

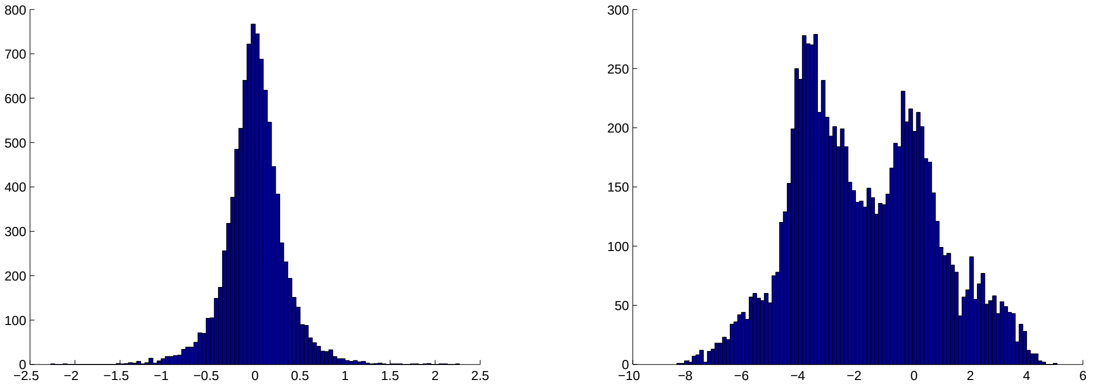


(a) A single realization of a Laplacian random walk with superimposed Gaussian noise. Dashed: Original signal. Solid: Noisy signal

(b) Reconstruction using the LSE estimate of the ROF model. Dashed: Original signal. Solid: LSE reconstruction

Figure 7.1: Reconstruction of a signal obtained from a noisy Laplacian random walk. The LSE estimate does not suffer from artifacts.

distribution. The residuals, however, only vaguely resemble a Gaussian distribution. Compared to the MAP results (Figure 3.2) , these result can still be considered as overall better.



(a) Distribution of the differences $x_{i+1} - x_i$ in the LSE reconstruction

(b) Distribution of the residuals

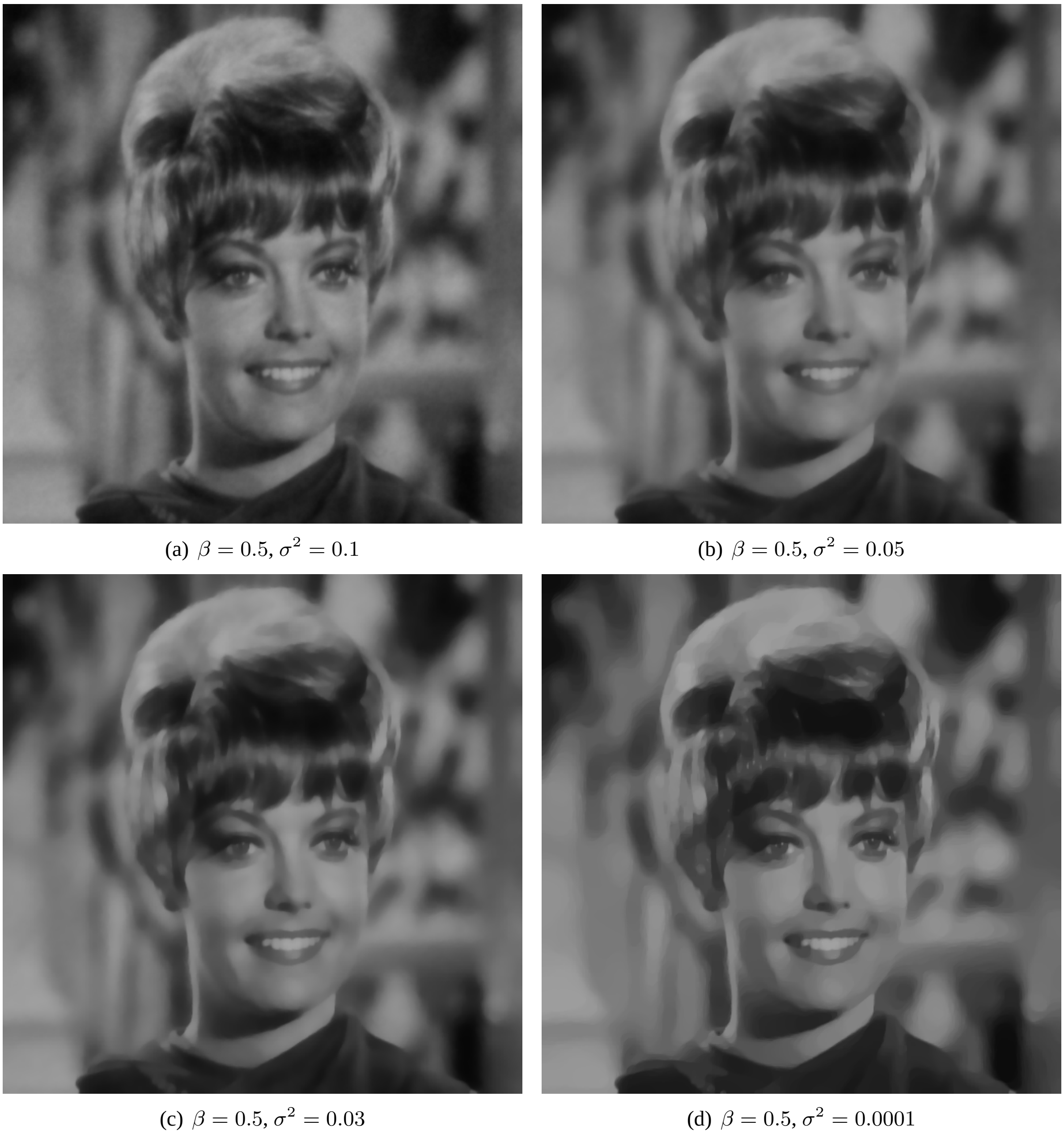
Figure 7.2: The distribution of the LSE estimate perfectly matches the desired Laplacian distribution (left). The distribution of the residuals does not resemble the true distribution that well (right). It is, however, more similar to the true distribution than the residual distribution of the MAP estimate.

Evaluation

The first part of this evaluation is concerned with the influence of the parameters on the denoising results.

Consider again the general form of the pdf that is induced by a variational model:

$$p(u|f) = \frac{1}{Z} \exp\left\{ -\frac{E(u;\lambda)}{T} \right\}$$



(a) $\beta = 0.5, \sigma^2 = 0.1$

(b) $\beta = 0.5, \sigma^2 = 0.05$

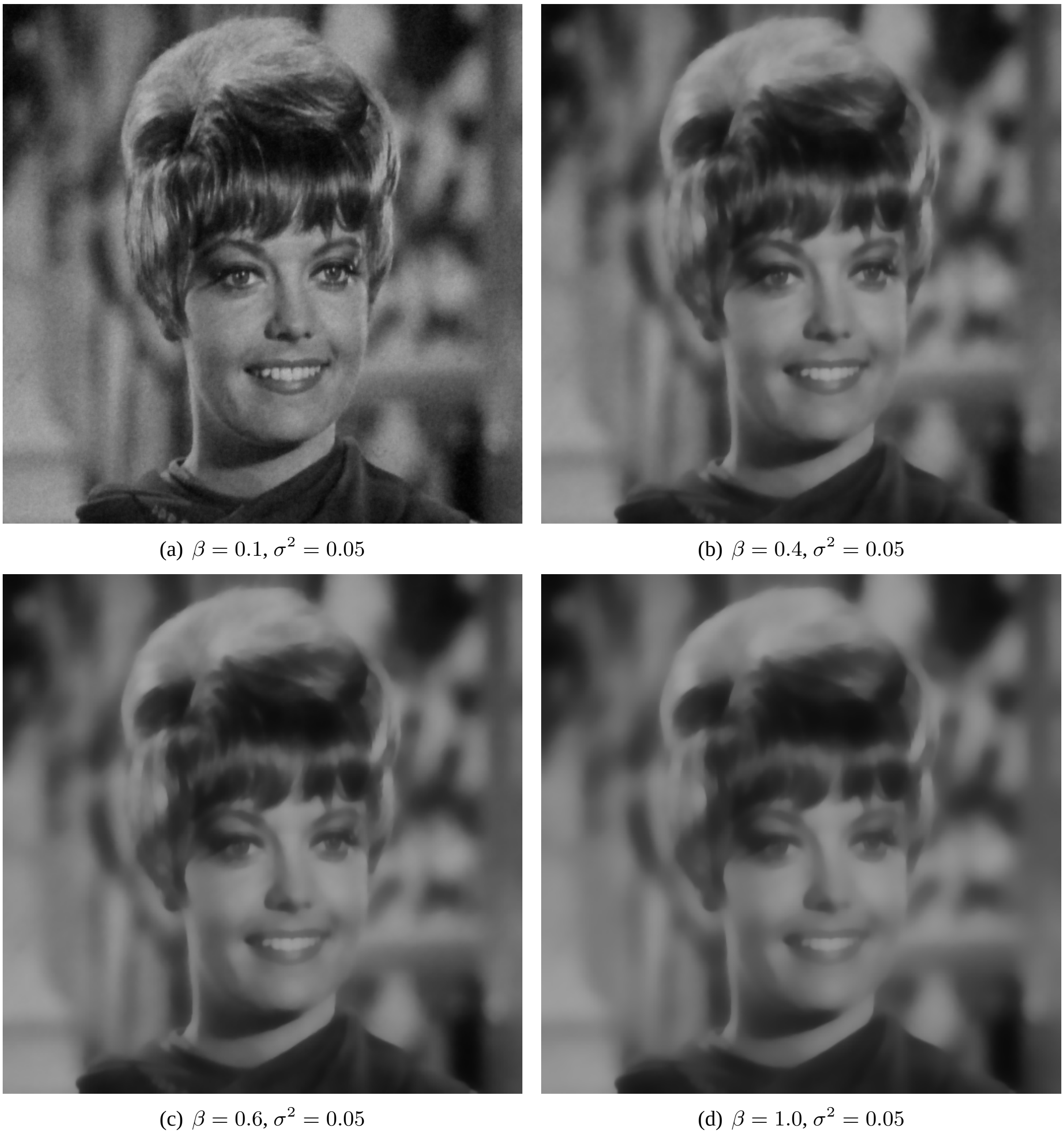
(c) $\beta = 0.5, \sigma^2 = 0.03$

(d) $\beta = 0.5, \sigma^2 = 0.0001$

Figure 7.3: Results of LSE denoising using the ROF model for fixed β and falling temperature. The temperature is given by $2\sigma^2$. As the temperature gets lower, the LSE denoiser approaches the MAP result.

The parameter λ controls the amount of regularization that is applied. The parameter T, called the temperature, controls the spread (variance) of the pdf. Let us examine the influence of this second parameter in the context of the ROF model (note that the principle results also apply to all other models that are considered in this thesis). In the ROF model, the temperature has a direct relation to the variance of the data term and is given by $T = 2\sigma^2$. Moreover, we use the regularization parameter $\beta = \frac{\lambda}{T}$ to emphasize the explicit dependence of the regularization parameter on the temperature in the LSE estimate.

Figure 7.3 shows how the denoising result changes for a fixed regularization parameter λ and different temperatures. Figure 7.4 shows the case where the temperature is fixed and λ is changed. All results were obtained from an image that was artificially degenerated by 10% Gaussian noise. From the first



(a) $\beta = 0.1$, $\sigma^2 = 0.05$

(b) $\beta = 0.4$, $\sigma^2 = 0.05$

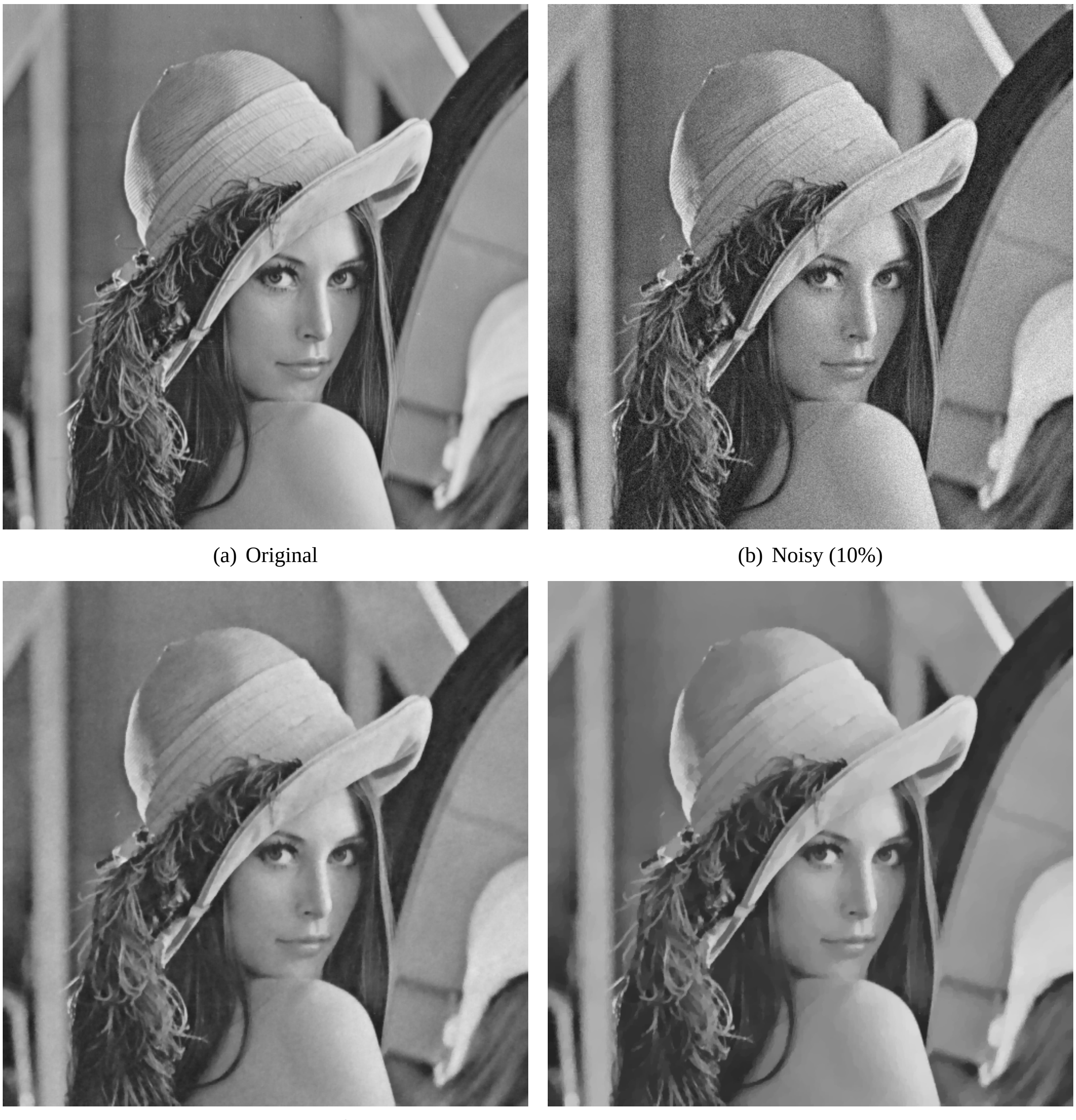
(c) $\beta = 0.6$, $\sigma^2 = 0.05$

(d) $\beta = 1.0$, $\sigma^2 = 0.05$

Figure 7.4: Results of LSE denoising using the ROF model for fixed σ^2 and rising β. The temperature is given by $2\sigma^2$. With rising β, the result becomes blurred.

experiment it is obvious that the results resemble the MAP estimate more and more as the temperature becomes lower. This can be attributed to the fact that as T becomes smaller, the variance of the probability distribution becomes smaller as well, which further leads to the effect that the maximal mode and the expected value approach each other. Conversely, if the temperature is held fixed, the LSE estimate becomes smoother, without introducing staircases. For λ very small or T very high the result approaches a noisy image, with a variance that depends on T. When both values are high the results becomes a blurred version of the cartoon that results from the MAP. Note that the temperature also influences the run-time of the algorithm. For larger T, the two chains in Algorithm 5 have to make larger leaps in the state space which results in a longer run-time.

Figures 7.5 and 7.6 shows a direct comparison of the denoising results of ROF-MAP and ROF-LSE respectively. The LSE reconstruction does not fully remove the noise. Depending on the combination of parameters λ and σ^2 there is always a noise residual in the reconstruction. In terms of visual quality, results from the LSE estimate tend to look more natural. This can be largely attributed to the absence of staircasing.



(a) Original

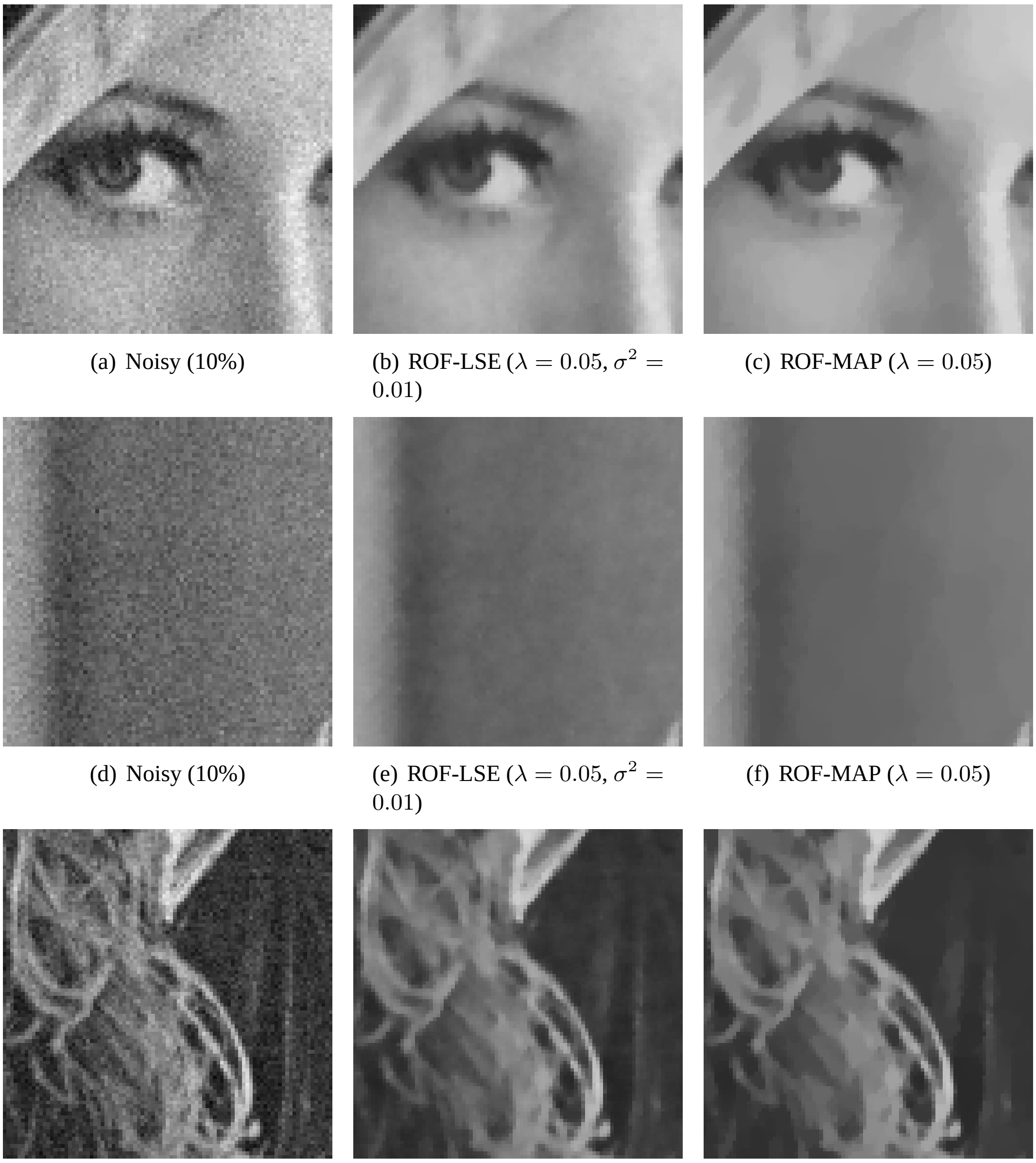
(b) Noisy (10%)

(c) ROF-LSE ($\lambda = 0.05$, $\sigma^2 = 0.01$)

(d) ROF-MAP ($\lambda = 0.05$)

Figure 7.5: Denoising using ROF-LSE and ROF-MAP. ROF-LSE leaves some noise in the reconstruction, but the overall result looks more natural due to the absence of staircasing.

For the quantitative comparison of ROF-MAP to ROF-LSE, we applied both denoisers with fixed parameters to the test images. The images where artificially degenerated by 4 different noise levels (5%, 10% and 20% additive white Gaussian noise). Table 7.1 shows the average SSIM on the test set. The regularization parameter λ is shown in parentheses. For the LSE denoiser σ^2 was fixed to 0.01. The quantitative comparison shows that ROF-LSE slightly outperforms the MAP denoiser in terms of average SSIM.



(a) Noisy (10%)

(b) ROF-LSE ($\lambda = 0.05$, $\sigma^2 = 0.01$)

(c) ROF-MAP ($\lambda = 0.05$)

(d) Noisy (10%)

(e) ROF-LSE ($\lambda = 0.05$, $\sigma^2 = 0.01$)

(f) ROF-MAP ($\lambda = 0.05$)

(g) Noisy (10%)

(h) ROF-LSE ($\lambda = 0.05$, $\sigma^2 = 0.01$)

(i) ROF-MAP ($\lambda = 0.05$)

Figure 7.6: Close-up of the denoised image. In the ROF-LSE reconstruction no staircases are visible.

Noise:	5% ($\lambda = 0.025$)	10% ($\lambda = 0.05$)	20% ($\lambda = 0.067$)	25% ($\lambda = 0.125$)
ROF-LSE	0.9204	0.8581	0.7866	0.7303
ROF-MAP	0.8973	0.8279	0.7833	0.7062

Table 7.1: Quantitative comparison of ROF-MAP to ROF-LSE in terms of average SSIM.

7.1.2 The TV-L1 Model

TV-L1 is another simple and popular model that was initially developed for the reconstruction of images that where degenerated by “Salt & Pepper” noise. The only difference to the ROF model is that the data term is given by the L1 norm (which comes from the fact that “Salt & Pepper” noise can be modeled using a Laplacian distribution). The energy of this model is given by:

$$E(u;\lambda) = \int_\Omega |u - f| dx + \lambda \int_\Omega |\nabla u| dx \tag{7.1}$$

TV-L1 can also be used for applications like structure-texture decomposition and shape denoising. Moreover the MAP estimate is contrast-invariant, which opens up the possibility to use the model for scale-driven feature selection [Pock, 2008].

Direct optimization of this model is problematic due to the fact that the absolute value is not continuously differentiable at zero (see section 2.2.2). Furthermore duality principles can not be applied here because the functional is only weakly convex. An approximation based on convex relaxation was proposed in [Aujol et al., 2006].

Using a probabilistic approach, we can directly optimize this model. The model is also unimodal, allowing us to employ the LSE estimator.

Evaluation

We again compare the denoising results of the MAP estimate (TV-L1-MAP) to the LSE estimate (TV-L1-LSE) on the 13 test images that where artificially degenerated by “Salt & Pepper” noise. The regularization parameter was fixed to $\lambda = 1$ for both models and all noise levels. For TV-L1-LSE the temperature was fixed to $T = 0.02$.

Figure 7.7 shows a direct comparison of the denoising results on an image that was heavily degenerated. Both denoisers are able to successfully reconstruct the image, despite the high level of noise. The difference between the results becomes more apparent in the close-up images that are provided in Figure 7.8. The LSE estimate successfully recovers details, where the MAP estimate fails. Moreover, similar effects to the ROF model can be observed. The LSE estimate leaves some noise in the image and does not suffer from staircasing artifacts.

Table 7.2 shows a quantitative comparison in terms of average SSIM, taken over all test images. The LSE estimate outperforms MAP at all noise levels. The difference in terms of SSIM is larger for TV-L1 when compared to ROF. Moreover we observe that the LSE estimator of the TV-L1 model needs roughly 20% less iterations to reach convergence compared to its ROF pendant. We believe that this effect can be attributed to the weak convexity of the model. Once both chains have reached the set of maximal points, most samples have the same probability (excluding occasional samples outside this set). This leads to a rapid convergence of the individual LSE estimates of the chains.

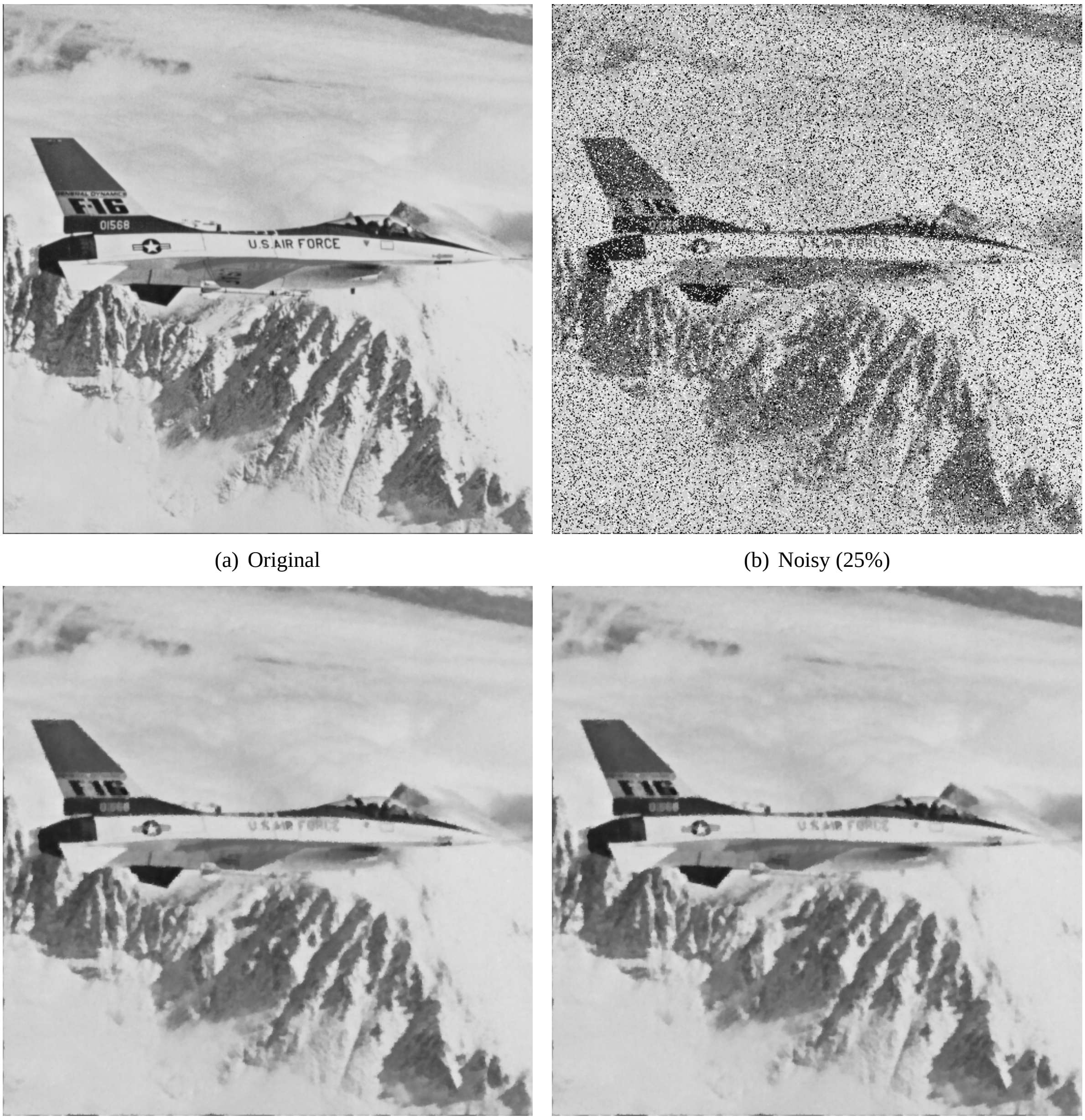
Noise:	5%	10%	20%	25%
TV-L1-LSE	0.8655	0.8534	0.8284	0.8144
TV-L1-MAP	0.8222	0.8112	0.7874	0.7752

Table 7.2: Quantitative comparison of TV-L1-MAP to TV-L1-LSE in terms of average SSIM.

Contrast invariance

The MAP estimate of the TV-L1 model exhibits an interesting feature [Chan and Esedoglu, 2005]: Given an image f and the corresponding result from MAP estimation u^*, for any contrast-adjusted image $c \cdot f$, with c scalar, it follows that $c \cdot u^*$ is a solution of the model for the contrast-adjusted image.

The effect can be used for scale-driven feature selection and is clarified in Figure 7.9, where for different values of the regularization parameter features with different scales vanish, regardless of their contrast.



(a) Original

(b) Noisy (25%)

(c) TV-L1-LSE ($\lambda = 1, T = 0.02$)

(d) TV-L1-MAP ($\lambda = 1$)

Figure 7.7: Denoising using TV-L1-LSE and TV-L1-MAP.

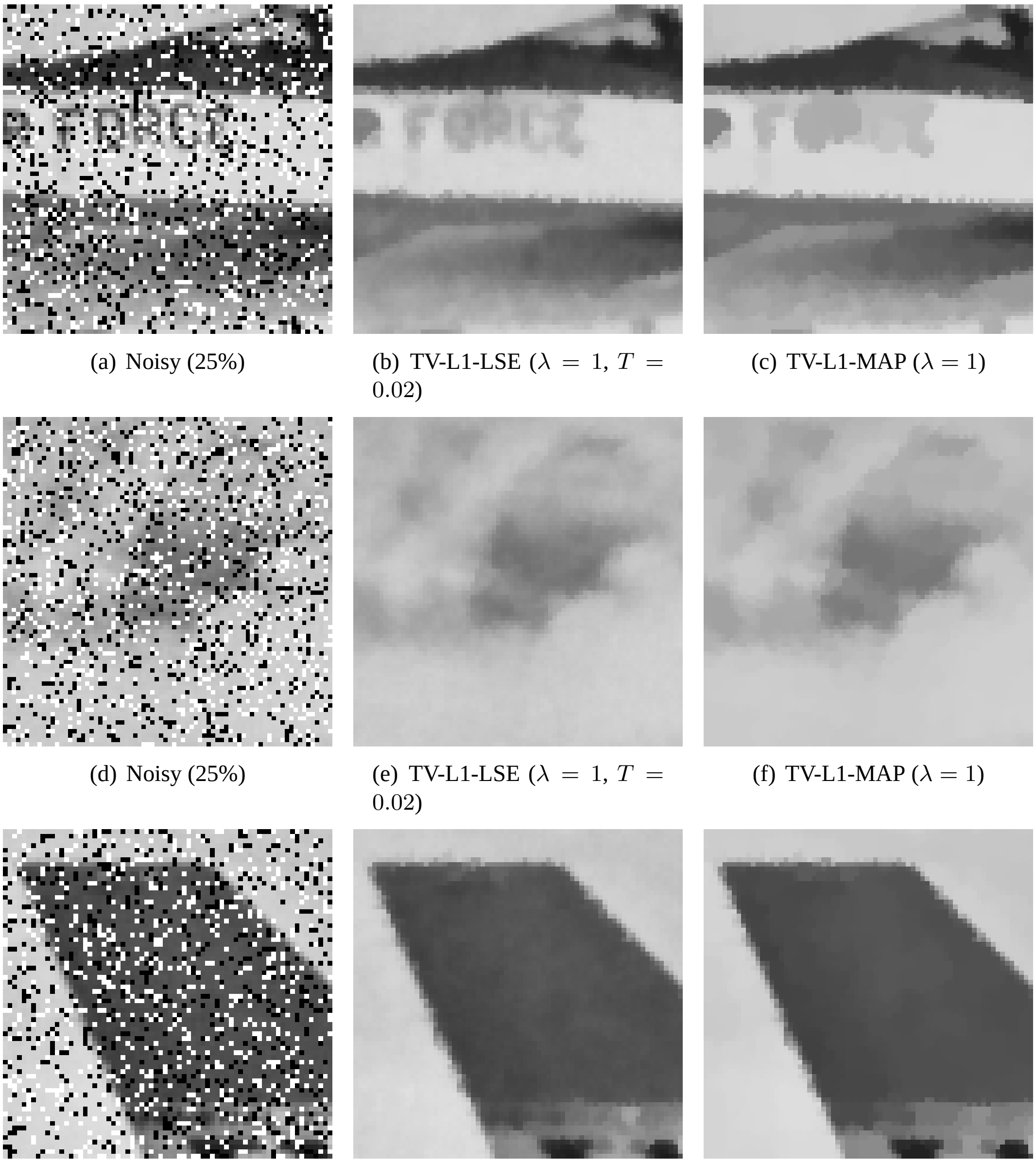
The contrast-invariance can be attributed to the dynamics of the model at a stationary point. The Euler-Lagrange equation of the model is given by

$$\frac{u^* - f}{|u^* - f|} - \lambda \nabla (\frac{\nabla u^*}{|\nabla u^*|}) = 0 \tag{7.2}$$

Now consider the same image with changed contrast $\hat{f} = c \cdot f$ and the solution $\hat{u}^* = c \cdot u^*$. Substituting these relations into (7.2) clearly shows that $\hat{u}^*$ actually is a solution of the modified Euler-Lagrange equation.

It seems considerably harder to obtain a similar analytic result for the LSE estimator. From our experiments it seems, however, that the LSE estimator for the TV-L1 model is contrast-invariant as well.

Figure 7.10 shows the pdf for two pixels. For different scalings c of the input data the LSE estimate is scaled by the same factor. Figure 7.11 shows the LSE estimates on a test image for different values



(a) Noisy (25%)

(b) TV-L1-LSE ($\lambda = 1$, $T = 0.02$)

(c) TV-L1-MAP ($\lambda = 1$)

(d) Noisy (25%)

(e) TV-L1-LSE ($\lambda = 1$, $T = 0.02$)

(f) TV-L1-MAP ($\lambda = 1$)

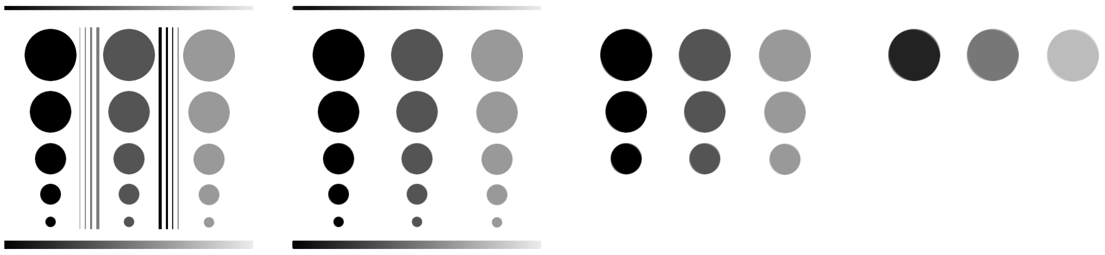
(g) Noisy (25%)

(h) TV-L1-LSE ($\lambda = 1$, $T = 0.02$)

(i) TV-L1-MAP ($\lambda = 1$)

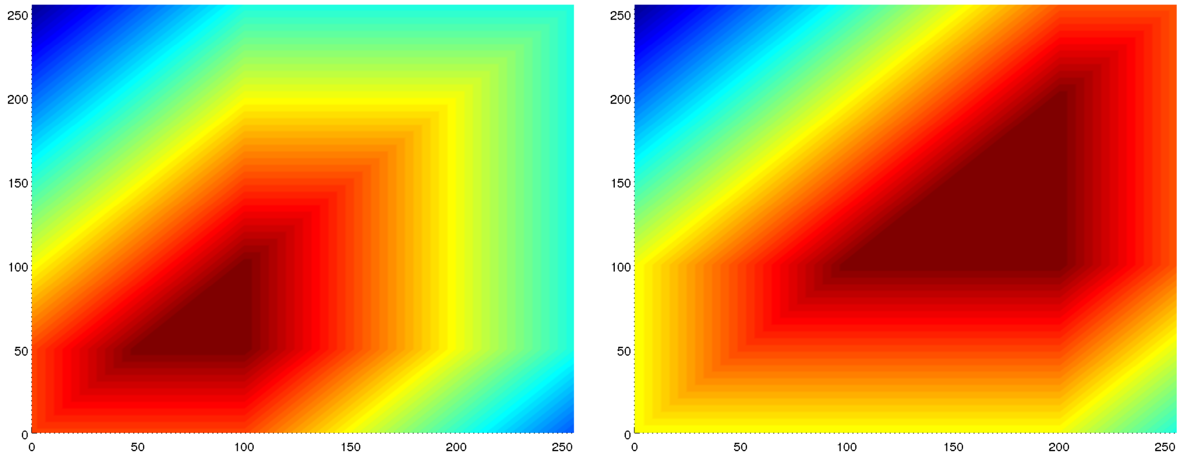
Figure 7.8: Close-up of the denoised image. Similar effects to the ROF model can be observed. TV-L1-LSE produces no staircases and leaves some noise in the denoised image.

of λ. Similar to the MAP estimate shown in Figure 7.9, different features vanish depending on the size of the feature, regardless of the specific contrast. Note, however, that larger features do not completely vanish. This can be attributed to the fact, that the LSE estimator is only approximated. For smaller values ϵ in Algorithm 5 the features tend to completely vanish. Note, however, that the effective run-time of the algorithm is strongly influenced by this parameter. Choosing ϵ too small results in a prohibitively long run-time of the algorithm, which effectively lessens the usefulness of the TV-L1-LSE estimator for scale-driven feature selection.



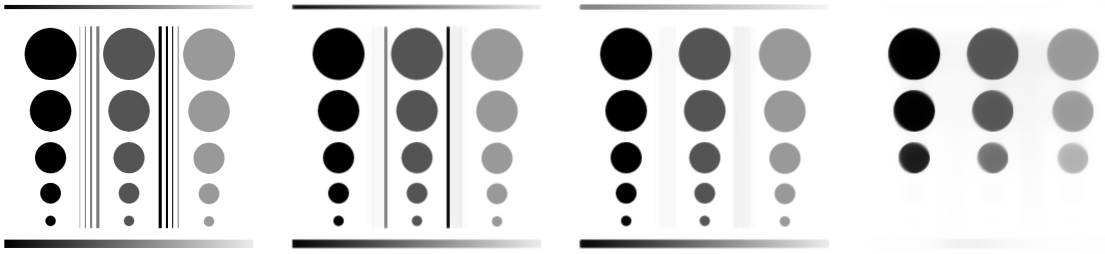
(a) Original image (b) $\lambda = 3.5$ (c) $\lambda = 12.0$ (d) $\lambda = 33.0$

Figure 7.9: Scale-driven feature selection using TV-L1-MAP. With stronger regularization larger features vanish, independent of their contrast.



(a) c = 1 (b) c = 2

Figure 7.10: Probability density function of the TV-L1 model for 2 pixels. The pixels were set to $f_1 = 100 \cdot c$ and $f_2 = 50 \cdot c$. The regularization parameter λ and the temperature T were fixed to 1. The resulting LSE estimates are $u = (83\frac{1}{3}, 66\frac{2}{3})$ for $c = 1$ and $u = (166\frac{2}{3}, 133\frac{1}{3})$ for $c = 2$.



(a) Original image (b) $\lambda = 1.5$ (c) $\lambda = 2.0$ (d) $\lambda = 10.0$

Figure 7.11: Scale-driven feature selection using TV-L1-LSE. With stronger regularization larger features vanish, independent of their contrast. The temperature was fixed to $T = 0.01$.

7.2 Estimating Motion

Variational models have been successfully used as the base of high-level algrithms that need accurate estimations of object motion between image frames.

From a low-level viewpoint the problem of motion estimation can be approached by estimating displacements of pixels. Such a procedure is generally refered to as optical flow estimation. Figure 7.12 shows an example along with the color-coded groundtruth flow field, where hue indicates the direction and saturation indicates the magnitude of the flow field.

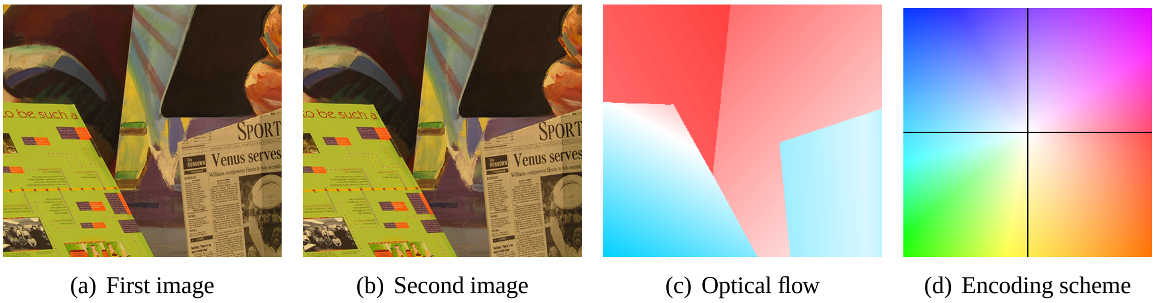

(a) First image (b) Second image (c) Optical flow (d) Encoding scheme

Figure 7.12: Example of optical flow estimation. The objective is to estimate pixel movement between the first image (a) and the second image (b). (c) shows the color-coded groundtruth flow. (d) shows the encoding scheme. Hue encodes direction while saturation encodes magnitude of the flow field.

Let us state the general variational formulation of the optical flow problem:

Given an image pair I_1, I_2 and a flow field $u = (u_x, u_y)^T : \Omega \to \mathbb{R}^2$, the variational energy is given as

$$E(u) = \mathcal{R}(u) + \mathcal{D}(u; I_1, I_2)$$

where again $\mathcal{R}(u)$ is a regularizer and $\mathcal{D}(u; I_1, I_2)$, the data term, measures pixel similarity between the displaced input images I_1 and I_2.

A simple data term is given by the so-called brightness-constancy assumption. We assume that a pixel retains its intensity when a movement occurs. Such an assumption of course is not robust with respect to illumination changes between images. In a seminal paper Horn and Schunck [1981] proposed a quadratic regularization of the gradients of the flow field together with a quadratic data term. This approach does not allow steep discontinuities in the estimated flow field and cannot robustly handle occlusions.

A better model is given by a total variation-based prior together with a L1 data term [Zach et al., 2007]. The energy of this model is given by

$$E(u; \lambda) = \int_\Omega |\nabla u_x|_\epsilon dx + \int_\Omega |\nabla u_y|_\epsilon dx + \frac{1}{\lambda} \int_\Omega |I_1(x+u) - I_2| dx \tag{7.3}$$

Note that the prior was sligthly modified. Instead of the usual L1 norm, the Huber norm

$$|q|_\epsilon = \begin{cases} \frac{q^2}{2\epsilon} & \text{if } |q| \leq \epsilon \\ |q| - \frac{\epsilon}{2} & \text{else} \end{cases}$$

is employed. This allows to penalize small variations in the gradient quadratically and avoids piecewise constant solutions. ϵ was set to 0.01 in all experiments.

The data term in (7.3) is non-convex, minimization of this model therefore is problematic. A typical approach is to approximate (7.3) by linearizing $I_y(x + u(x))$ around some initial displacement u_0 using a Taylor expansion:

$$I_1(x+u) \approx I_1(x+u_0) + \langle u - u_0, \nabla I_1 \rangle \tag{7.4}$$

By substituting (7.4) into the energy (7.3), a weakly convex approximation of the initial energy is obtained. The linearization however poses a problem: It is only valid for small displacements around u_0. Optimization of the relaxed energy is therefore embedded within a multilevel warping scheme, where an image pyramid is build from the input images and minimization is first carried out on the coarsest level and subsequently propagated to the next level. This procedure is repeated until the base of the pyramid (i.e. the original input images) has been reached.

7.2.1 Stereo Reconstruction

We first consider the simplified case, where the flow field has only one component, i.e.

$$u = (u_x, 0)^T$$

Given a rectified stereo image pair the estimated flow u_x is a 2.5D depthmap of the depicted scene.

Using the sampling-based MAP estimator (Algorithm 6), the energy (7.3) can be minimized directly. No linearization is needed because the energy can be globally optimized. A multilevel approach is still necessary, however, because the sampler tends to have low local acceptance rate in untextured regions. Moreover a faster cooling schedule can be employed if a multilevel approach is used, because the schedule is directly related to the the scanned disparity ranges.

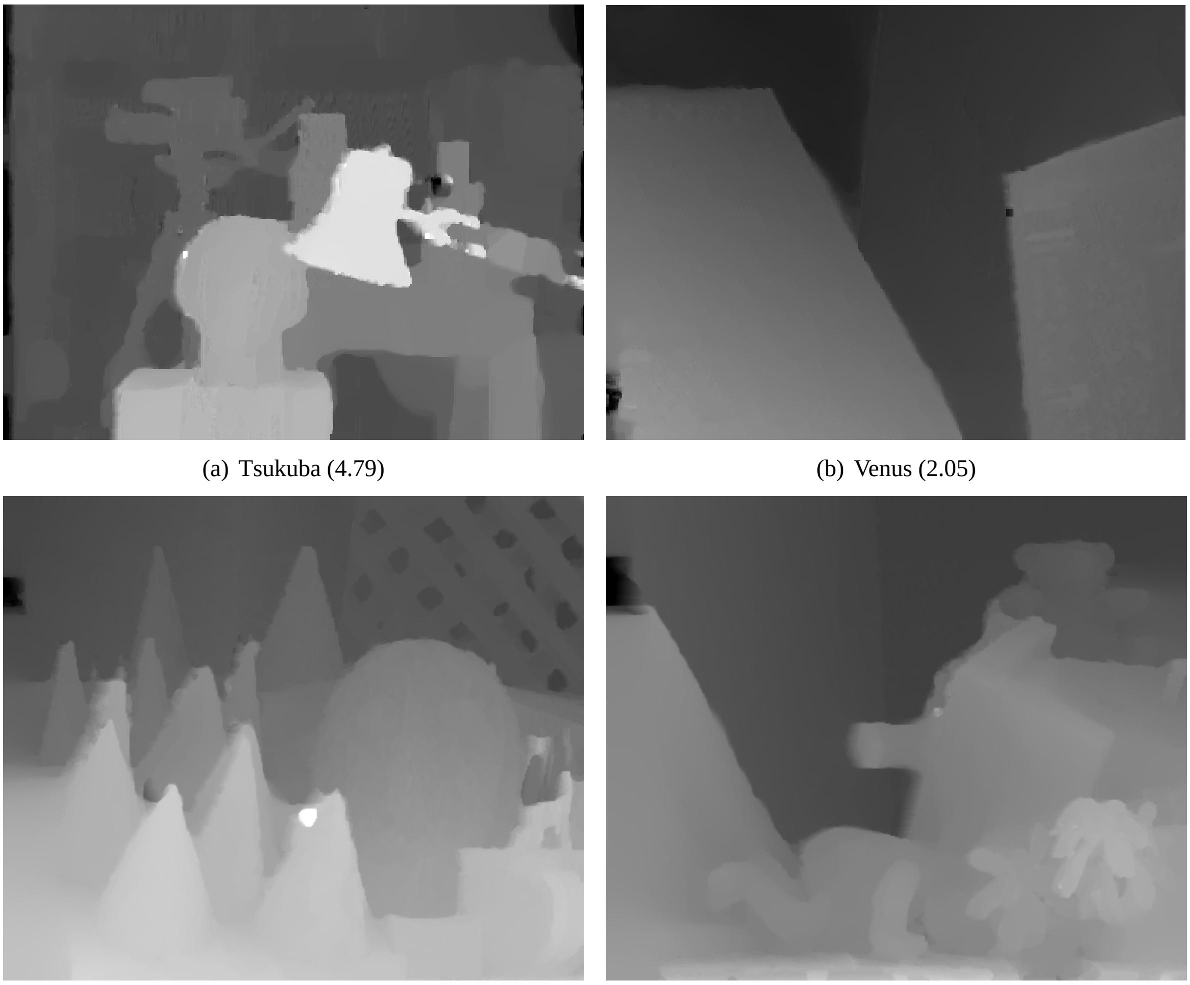
To achieve results that do not get stuck in local minima, some considerations have to be made: First the number of Metropolis-Hastings iterations per temperature step K is crucial. If the number is too small the algorithm gets stuck in local minima. If the value is too large, the algorithms takes a prohibitively long runtime. Local optima are approached in the early stages of the algorithm, where the temperature is still high. This is simply explained by the fact that sampler can make large jumps in this stage (and therefore the adapted variance α_k of the proposal distribution is large). The probability of hitting just the right mode of the target distribution in only a few iterations is therefore relatively low. A simple adaption of the parameter K has shown to provide stable results. We choose $K = \lceil 2\alpha_k \rceil + 2$, i.e. we make the number of Metropolis-Hastings iterations explicitely dependent on the spread of the proposal distribution. For low spread (which corresponds to low temperatures) we make at least 3 Metropolis-Hastings steps. For high spread the sampler is allowed to make more moves before the temperature is lowered.

Second, the initial temperature has to be high enough, to allow the sampler to move freely in the early stages of the algorithm. For all our experiments we chose $T_0 = 5$. This has shown to provide good results on the test dataset. Note that for scenarios with small disparities one can choose a lower initial temperature to speed up the algorithm.

Last, the algorithm needs to run long enough to reach a sufficiently low temperature. This is crucial for the quality of the results, as the small scale details are infered at low temperatures. We chose to stop the algorithm if the energy change between 100 successive moves is small enough (≈ 1). This allows a sufficiently low temperature at the later stages of the algorithm.

The proposed algorithm was evaluated on 4 image sequences from the Middlebury stereo evalution database [Scharstein and Szeliski, 2003] and compared to the results from the respective linearization approach (computed using the PDHG algorithm that was introduced in chapter 2). We will further refer to the sampling-based approach as TV-L1-SA and the linearization approach as TV-L1-PD. To allow a direct comparison, the regularization parameter was fixed to $\lambda = 30$ for all experiments.

Figure 7.13 shows the results of TV-L1-SA, along with the percentage of wrongly labeled pixels (i.e. where the disparity error is larger than one). Figure 7.14 shows a direct comparison between the results of TV-L1-SA and TV-L1-PD on the "Cones" sequence along with an error map (white = correct label, black = error, gray = occlusion). The global optimization approach is able to correctly infere small-scale structure, where the linearized model fails.



(a) Tsukuba (4.79)

(b) Venus (2.05)

(c) Teddy (12.05)

(d) Cones (6.91)

Figure 7.13: Stereo reconstruction using TV-L1-SA. The percentage of wrongly labeled pixels is shown in parentheses

The similarity measure used so far is very simplistic and not very robust. The model can be refined by replacing the data term with a more sophisticated similarity measure.

A more robust measure can be formulated, if similarity is based on patches of pixels instead of single pixels. Werlberger et al. [2010] propose to use a truncated normalized crosscorrelation (TNCC) for the similarity matching:

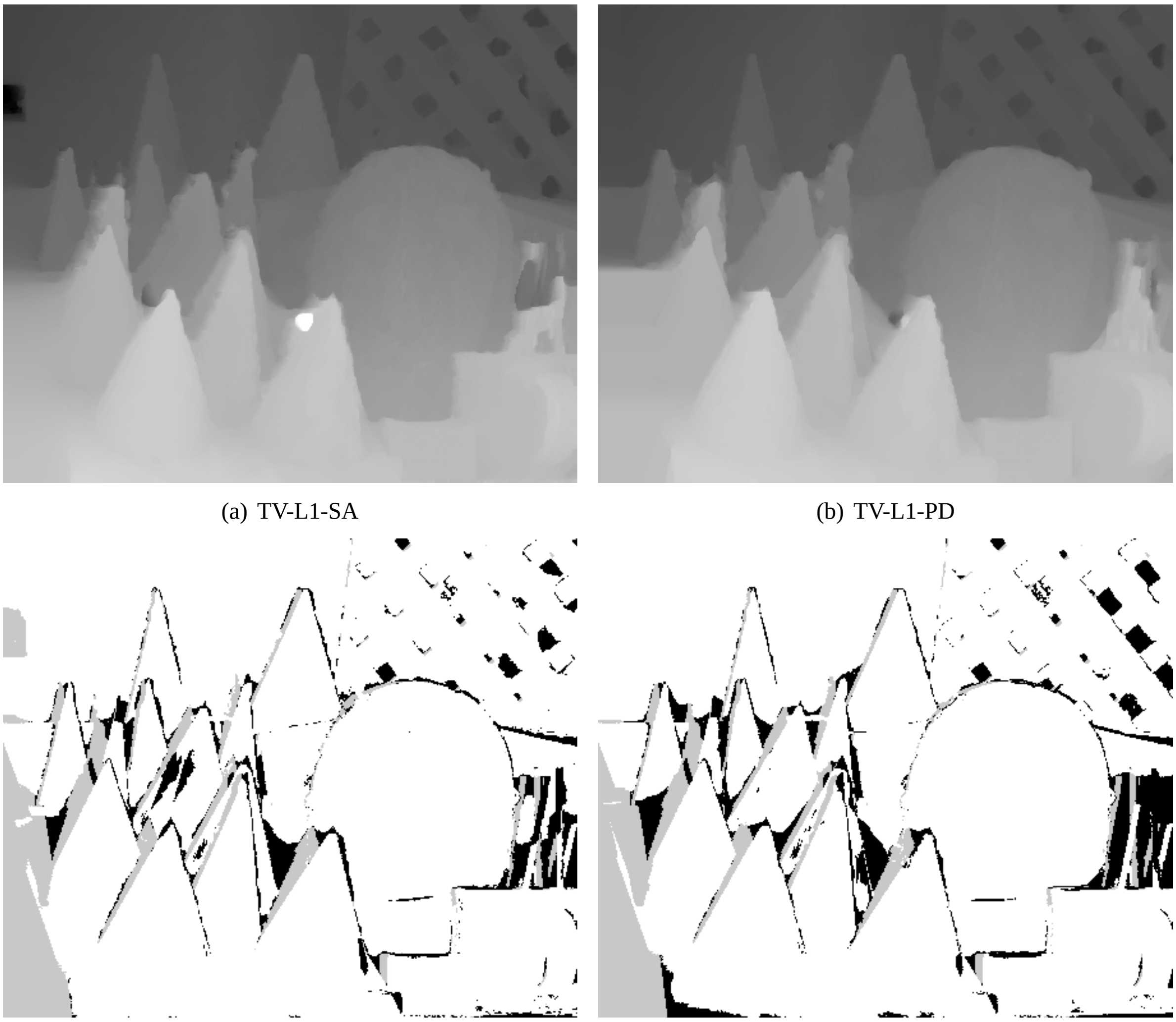
Let $B_W(x-y)$ denote a box filter of width W and $\int_\Omega B_w(z)dz = 1$. Then the means of a patch centered at x for the images $\hat{I}_1 = I_1(x+u)$ and I_2 are given by

$$\mu_1(x) = \int_\Omega \hat{I}_1(y) B_W(x-y)dy \qquad \mu_2(x) = \int_\Omega I_2(y) B_W(x-y)dy$$

and the variances of the patch are given by

$$\sigma_1(x) = \int_\Omega (\hat{I}_1(y) - \mu_1(x))^2 B_W(x-y)dy$$
$$\sigma_2(x) = \int_\Omega (I_2(y) - \mu_2(x))^2 B_W(x-y)dy$$

Using the definitions above, the normalized crosscorrelation between the images $\hat{I}_1$ and I_2 at location x


(a) TV-L1-SA

(b) TV-L1-PD

(c) Error (TV-L1-SA)

(d) Error (TV-L1-PD)

Figure 7.14: Comparison of TV-L1-SA to TV-L1-PD on the “Cones” sequence. The error map (white = correct, black = error, gray = occlusion) shows that TV-L1-SA provides better results than TV-L1-PD in regions where small-scale structure is present.

then reads

$$NCC(x,u)) = \frac{1}{\sqrt{\sigma_1(x)\sigma_2(x)}} \cdot \int_\Omega (\hat{I}_1(y) - \mu_1(x))(I_2(y) - \mu_2(x)) B_W(x-y) dy$$

At last the NCC is truncated to only allow for positive correlations. This results in the following data term:

$$D(u; I_1, I_2) = \int_\Omega \min(1, 1 - NCC(x,u)) dx \tag{7.5}$$

The resulting functional is again non-convex and moreover highly nonlinear. Werlberger et al. [2010] propose a second-order Taylor expansion to obtain a convex approximation of the TNCC data term (further refered to as TV-NCC-PD). Using Simulated Annealing we can again directly solve the non-convex optimization problem.

For the NCC data term, we observe an interesting effect: Smaller regularization is problematic, the algorithm tends to produce bad results in strongly textured regions (i.e. in regions where the TNCC term can be expected to be high). To get acceptable results the regularization has to be chosen higher. Slower annealing schedules or extending the multilevel procedure to coarser levels do not change this effect. We therefore chose $\lambda = 2$ for all experiments.

Figure 7.15 shows the results that where obtained by the proposed algorithm. In the TV-NCC approach the Tsukuba dataset is problematic due to a cluttered background.

Figure 7.15(e) shows a result of the TV-NCC-PD algorithm for the “Teddy” sequence for direct comparison. Details like the teddy bear in the upper right corner are far better reconstructed by the proposed algorithm.

Finally, a quantitative comparison is shown in Table 7.3. The numbers show that global optimization has a huge potential. Even the simple TV-L1 model yields very good results when solved directly. The TNCC data term further lowers the error.

Dataset:	Tsukuba	Venus	Teddy	Cones	Average
TV-L1-SA	**4.79**	2.05	12.05	6.91	6.45
TV-L1-PD	5.97	3.69	16.4	9.10	8.79
TV-NCC-SA	5.31	**1.41**	**8.36**	**5.43**	**5.13**
TV-NCC-PD	5.80	1.77	14.2	8.42	7.55

Table 7.3: Quantitative comparison of the presented algorithms. The error is given as the percentage of wrongly labeled pixels. Simulated Annealing outperforms the respective PD algorithms. Note that the simpler TV-L1 model even outperforms TV-NCC if it is solved directly.

7.2.2 Optical Flow

In our last experiment, the total variation is replaced by a more sophisticated regularity measure. In [Werlberger et al., 2010] a novel regularizer, called non-local total variation, was proposed. The idea of this regularization is that pixel interactions are not constrained to direct neighbors, but also larger patches of pixels are allowed to interact.

Non-local total variation is defined as follows:

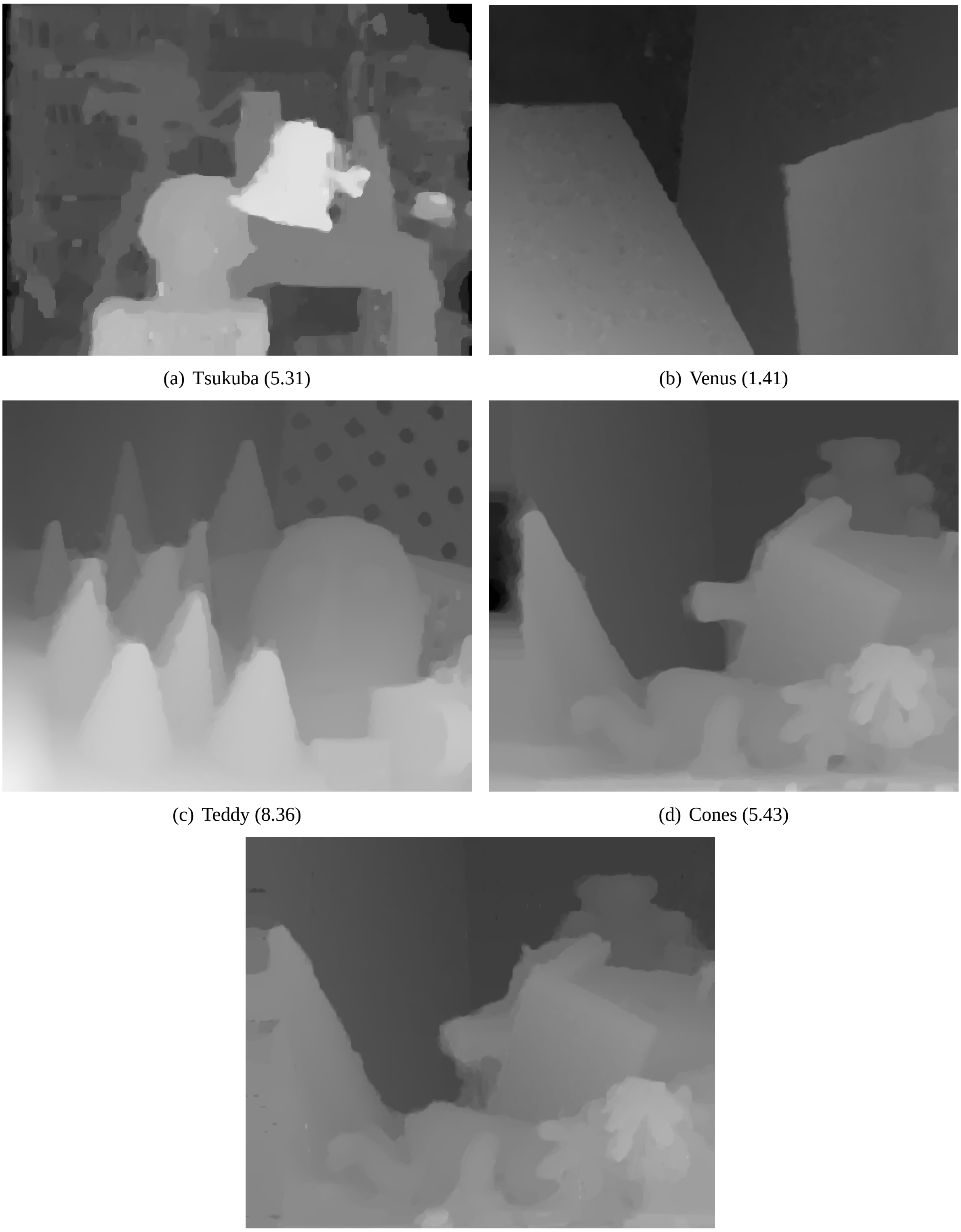
$$\mathcal{R}(u) = \int_\Omega \int_\Omega w(x,y)(|u_x(x) - u_x(y)|_\epsilon) + |u_y(x) - u_y(y)|_\epsilon)dydx \tag{7.6}$$

The term $w(x, y)$ weights the influence between the motion vectors at the positions x and y respectively and is given by

$$w(x,y) = \exp\left(-\left(\frac{\Delta_c(x,y)}{\alpha} + \frac{\Delta_s(x,y)}{\beta}\right)\right)$$

In the original formulation, $\Delta_c(x, y)$ measures color similarity in the Lab colorspace between the pixels at positions x and y. Our implementation does not use color information, the similarity measure is therefore simply given by the L1 norm of the illumination difference. The term $\Delta_s(x, y)$ measures the Euclidean distance between pixels. α and β can be used to tune the influence of both terms. Note that (7.6) in principle allows interaction between every pixel in the image. To constrain the computational complexity of the regularizer, Werlberger et al. [2010] recommend to have non-zero weights only in a window of size $2\beta + 1$ around a pixel. No essential information is lost with such a constraint, because the proximity influence of pixels outside of this window vanishes.

The experiments were carried out on 8 image pairs from the Middlebury optical flow dataset [Baker et al., 2007], were the images have flow components in both directions, i.e. $u = (u_x, u_y)^T$. For the experiments the NCC similarity measure was used. We again compare the global optimization approach (NLTV-NCC-SA) to the results from the convex second-order expansion that was obtained using a PDHG



(a) Tsukuba (5.31)

(b) Venus (1.41)

(c) Teddy (8.36)

(d) Cones (5.43)

(e) Teddy: TV-NCC-PD

Figure 7.15: Stereo reconstruction using TV-NCC-SA. The percentage of wrongly labeled pixels is shown in parentheses. (e) shows the reconstruction obtained from TV-NCC-PD. Details like the teddy bear in the upper right corner are clearly better reconstructed by the TV-NCC-SA algorithm.

algorithm (NLTV-NCC-PD). The parameters were fixed to $\alpha = 2$, $\beta = 5$ and $\lambda = 3$. For the Simulated

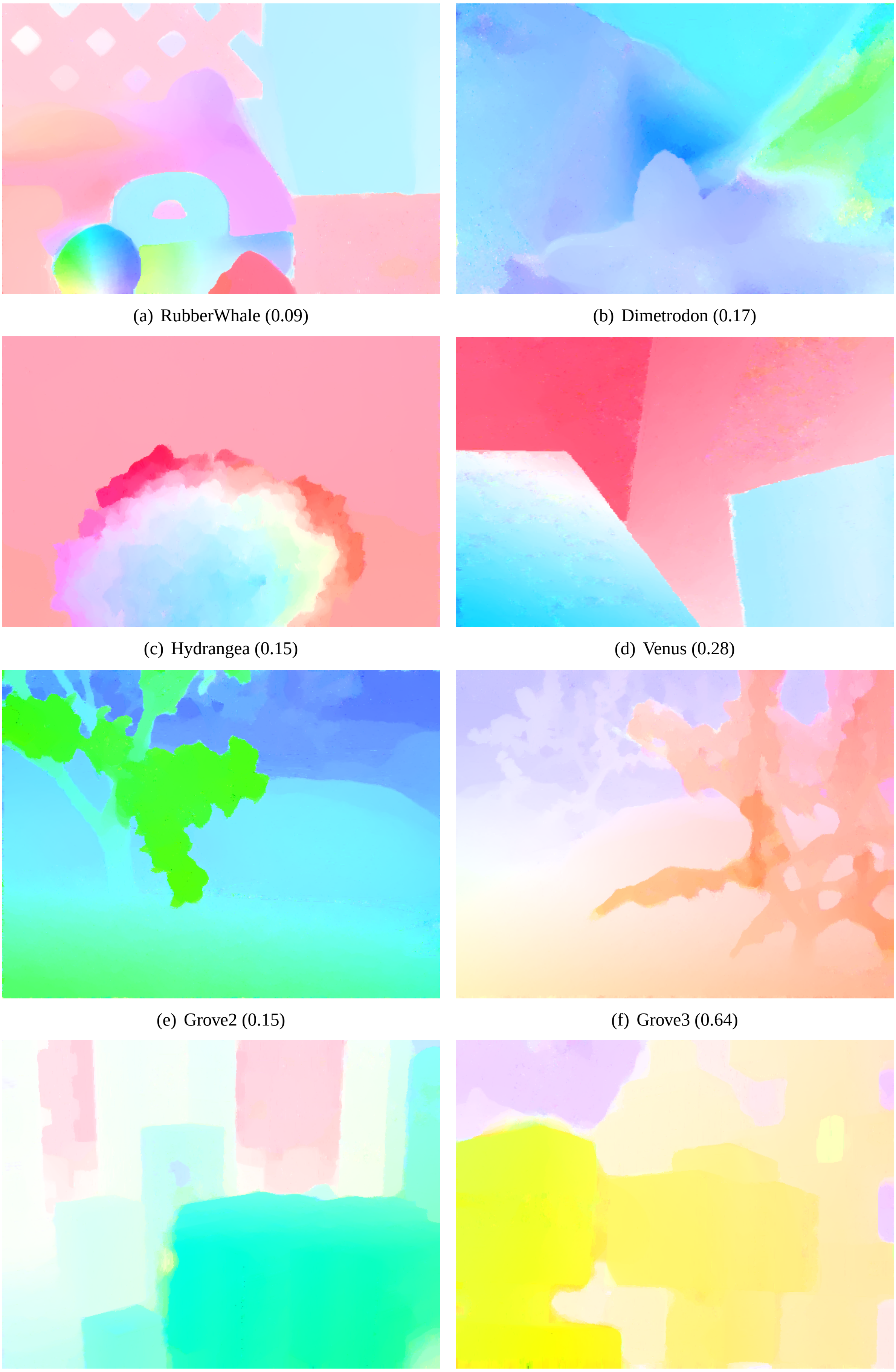
Annealing algorithm we again chose an initial temperature $T_0 = 5$. As error metric we report the average endpoint error (AEPE) that measures the average Euclidean distance between the estimated flow field and the ground truth.

Table 7.4 shows the quantitative results for both algorithms. Similar to the stereo case the global optimization approach constantly yields better results.

Model:	NLTV-NCC-PD	NLTV-NCC-SA
RubberWhale	0.12	0.09
Dimetrodon	0.20	0.17
Hydrangea	0.18	0.15
Venus	0.29	0.28
Grove2	0.19	0.15
Grove3	0.75	0.64
Urban2	0.40	0.38
Urban3	0.66	0.62

Table 7.4: Quantitative comparison of NLTV-NCC-SA to NLTV-NCC-PD in terms of average endpoint error.

Finally, Figure 7.16 shows the obtained flow fields as color-coded images. It can again be observed that the algorithm can successfully estimate flow, where small-scale features are present.



(a) RubberWhale (0.09)

(b) Dimetrodon (0.17)

(c) Hydrangea (0.15)

(d) Venus (0.28)

(e) Grove2 (0.15)

(f) Grove3 (0.64)

(g) Urban2 (0.38)

(h) Urban3 (0.62)

Figure 7.16: Estimated flow fields using NLTV-NCC-SA.

Chapter 8

Conclusion and Outlook

In this thesis, it was shown that an alternative approach to variational models in Computer Vision can provide several advantages.

We have shown that the energy minimization approach is equivalent to a stochastic procedure, known as MAP estimation. MAP is a simple point estimate in the posterior probability distribution, not taking into account specific characteristics of the underlying pdf. The insufficient use of information in the posterior distribution leads to distortions in the result, which are known in the context of image reconstruction as staircases. Typical approaches to mitigate these distortions consist in the design of more complicated priors.

The use of a simple summary statistic, the expected value, that compresses more information about the posterior distribution into a single estimate has shown to provide better results without modification of the original model.

The estimation of such a summary statistic poses some computational challenges. Inference of the expected value in principle requires knowledge of all possible images along with their posterior probabilities. Such a computation is infeasible even for small images.

To approximate the expected value, nonetheless, we have introduced MCMC algorithms, which are able to sample arbitrary high-dimensional distributions. We introduced the reader to the theory of general state space Markov Chains, which perfectly match the continuous paradigm that is employed in variational models. Different variations of MCMC sampling algorithms were discussed and led to a procedure which is well adapted to the needs of image processing.

Not all models are directly suitable for such a modification to the estimation scheme. Non-convex models tend to have an expected value that itself has very low probability in the model. Such an estimation seldom yields satisfactory results. Note however that such non-convex energies are even in the energy minimization approach not solved directly. Typically, the non-convex model is approximated by a convex model. The probabilistic point of view can also provide some advantages in this scenario. The use of a sampler provides us with knowledge about the posterior distribution which can also be used for MAP inference. Using a modified sampling scheme non-convex energies can be directly optimized without going the often difficult route of a convex approximation.

Our experiments show that our approach indeed is viable. Using different denoising models, we demonstrated the superiority of the LSE estimator over the usual MAP inference. The second application demonstrated the use of sampling algorithms for MAP inference using different optical flow models. Again this procedure has shown to be an alternative to the usual energy minimization approaches.

Last but not least we like to emphasize the generality of our approach. The presented Metropolis-Hastings sampling algorithm is suited for any probability density function, allowing to implement a wide range of models in a simple and fast way. While our approaches are not yet real-time capable (in contrast to numerous energy minimization applications), we see its main advantage in the rapid prototyping and testing of new models.

8.1 Outlook

The idea to use Bayesian estimators in the context of variational models instead of the usual energy minimization approach is relatively young. Therefore, there are numerous directions for further work.

From the viewpoint of estimation theory a lot of different loss functions are possible. This thesis was only concerned with “hit-or-miss” loss (for MAP inference) and a quadratic loss function (leading to the LSE estimator). Both losses are simplistic and quite general, there is, however, no reason not to design estimators based on more sophisticated loss functions. As an example we can consider non-convex energies: An estimator that combines ideas from LSE estimation and MAP estimation could be clearly advantageous. Instead of estimating the largest mode or the center of mass of the whole pdf, one could estimate the center of mass of the largest mode. If this mode has a heavy tail, such an estimate would probably outperform the MAP estimate.

The algorithmic side of this thesis was solely concerned with a sampling based approach. The sampling algorithms are generally known to be relatively slow. A speed-up could be achieved by using a proposal distribution that better matches the target distribution. As a rule of thumb, the proposal distribution should be as similar to the target distribution as possible (with the extremal value that both are equal). A speed-up using approximations of the target pdf together with a sampling scheme seems therefore possible. Algorithms which allow free form approximations, i.e. where no assumptions on the form of the approximation have to be made, were developed in recent years, with its two most notable instances Variational Bayes [Beal, 2003] and Expectation Propagation [Minka, 2001]. Both algorithms are general enough to account for the variety of different variational models, and hence seem as a good starting point for a refinement of the sampling scheme.

Another route would be to more closely examine Langevin diffusions (and similar stochastic differential equations). Eventually, one could find a stable ULA scheme that does not require the intermediate Metropolis-Hastings step. This approach seems promising for real-time applications.

Finally, the sampling algorithms could be subject to further optimization. Current research for the Metropolis-Hastings-based samplers seems to go into the direction of more sophisticated adaption schemes, leading to more efficient samplers.

Bibliography

Arnold, D. (2001). *A concise introduction to numerical analysis (Lecture notes)*. University of Minnesota, Minneapolis.

Atchadé, Y. (2006). An adaptive version for the Metropolis adjusted Langevin algorithm with a truncated drift. *Methodology and Computing in Applied Probability*, 8(2):235–254.

Aujol, J., Gilboa, G., Chan, T., and Osher, S. (2006). Structure-Texture Image Decomposition - Modeling, Algorithms, and Parameter Selection. *International Journal of Computer Vision*, 67(1):111–136.

Aujol, J. and Kang, S. (2006). Color image decomposition and restoration. *Journal of Visual Communication and Image Representation*, 17(4):916–928.

Baker, S., Roth, S., Scharstein, D., Black, M. J., Lewis, J., and Szeliski, R. (2007). A Database and Evaluation Methodology for Optical Flow. *2007 IEEE 11th International Conference on Computer Vision*, (October):1–8.

Beal, M. (2003). *Variational algorithms for approximate Bayesian inference*. PhD thesis, University College London.

Bélisle, C. (1992). Convergence theorems for a class of simulated annealing algorithms on R d. *Journal of Applied Probability*, 29(4):885–895.

Blomgren, P. and Chan, T. (1998). Color TV: total variation methods for restoration of vector-valued images. *IEEE Transactions on Image Processing*, 7(3):304–309.

Blum, L., Blum, M., and Shub, M. (1986). A simple unpredictable pseudo-random number generator. *SIAM Journal on Computing*, 15(2):364–383.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Bredies, K., Kunisch, K., and Pock, T. (2009). Total generalized variation. *SIAM Journal on Imaging Sciences*,.

Canal, L. (2005). A normal approximation for the chi-square distribution. *Computational Statistics & Data Analysis*, 48(4):803–808.

Caselles, V., Chambolle, A., and Novaga, M. (2008). The discontinuity set of solutions of the TV denoising problem and some extensions. *Multiscale Modeling and Simulation*, 6(3):879–894.

Chambolle, A. (2004). An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20(1):89–97.

Chambolle, A. and Pock, T. (2010). A first-order primal-dual algorithm for convex problems with applications to imaging. *Preprint*, pages 1–49.

Chan, T., Esedoglu, S., and Park, F. (2007). Image decomposition combining staircase reduction and texture extraction. *Journal of Visual Communication and Image Representation*, 18(6):464–486.

Chan, T., Esedoglu, S., Park, F., and Yip, A. (2005). Recent developments in total variation image restoration. In *In Mathematical Models of Computer Vision*, pages 1–18. Springer Verlag.

Chan, T. and Wong, C. (2002). Total variation blind deconvolution. *Image Processing, IEEE Transactions on*.

Chan, T. F. and Esedoglu, S. (2005). Aspects of Total Variation Regularized L1 Function Approximation. *Siam Journal of Applied Mathematics*, 65(5):1817.

Esser, J. (2010). *Primal Dual Algorithms for Convex Models and Applications to Image Restoration, Registration and Nonlocal Inpainting*. PhD thesis, University of California.

Fessler, J. a. (1996). Mean and variance of implicitly defined biased estimators (such as penalized maximum likelihood): applications to tomography. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 5(3):493–506.

Fisher, R. (1922). On the interpretation of χ 2 from contingency tables, and the calculation of P. *Journal of the Royal Statistical Society*, 85(1):87–94.

Geman, S. and Geman, D. (1993). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *Journal of Applied Statistics*, 20(5):25–62.

Grenander, U. and Miller, M. (1994). Representations of knowledge in complex systems. *Journal of the Royal Statistical Society*, 56(4):549–603.

Haario, H., Saksman, E., and Tamminen, J. (2001). An Adaptive Metropolis Algorithm. *Bernoulli*, 7(2):223.

Hastings, W. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

Horn, B. and Schunck, B. (1981). Determining optical flow. *Artificial intelligence*, 17(1-3):185–203.

Huang, J. and Mumford, D. (1999). Statistics of natural images and models. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1:541–547.

Katafygiotis, L. and Zuev, K. (2008). Geometric insight into the challenges of solving high-dimensional reliability problems. *Probabilistic Engineering Mechanics*, 23:208–218.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science (New York, N.Y.)*, 220(4598):671–80.

Lansel, S. (2007). DenoiseLab Philosophy: A Standard Test Set and Evaluation Method to Compare Denoising Algorithms. *http://www.stanford.edu/˜slansel/DenoiseLab*.

Louchet, C. (2008). *Variational and Bayesian models for image denoising: from total variation toward non-local means*. PhD thesis, Université Paris Descartes.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., and Teller, A. (1953). Equations of State Calculations by Fast Computing Machines, 1953. *Journal of Chemical Physics*, 21(6):1087–1092.

Meyn, S. and Tweedie, R. (1993). *Markov chains and stochastic stability*. Springer-Verlag.

Minka, T. (2001). Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Converence in Uncertainty in Artificial Intelligence*, pages 362–369.

Neal, P. and Roberts, G. (2006). Optimal scaling for partially updating MCMC algorithms. *Annals of Applied Probability*, 16(2):475515.

Nesterov, Y. (2004). *Introductory Lectures on Convex Optimization: A Basic Course (Applied Optimization)*. Springer Netherlands, 1 edition.

Nikolova, M. (2000). Local Strong Homogeneity of a Regularized Estimator. *SIAM Journal on Applied Mathematics*, 61(2):633.

Nikolova, M. (2004). Weakly Constrained Minimization: Application to the Estimation of Images and Signals Involving Constant Regions. *Journal of Mathematical Imaging and Vision*, 21(2):155–175.

Nikolova, M. (2007). Model distortions in Bayesian MAP reconstruction. *Inverse Problems and Imaging*, 1(2):399.

Nvidia (2009). NVIDIA CUDA C Programming Best Practices Guide CUDA Toolkit 2.3. (July).

Pock, T. (2008). *Fast Total Variation for Computer Vision*. PhD thesis, Graz University of Technology.

Pock, T., Cremers, D., Bischof, H., and Chambolle, A. (2009). An Algorithm for Minimizing the Mumford-Shah Functional. *Science*, (813396).

Ring, W. (2000). Structural properties of solutions to total variation regularization problems. *M2AN Mathematical modelling and numerical analysis*, 24:799–810.

Roberts, G. and Rosenthal, J. (2009). Examples of adaptive MCMC. *Journal of Computational and Graphical Statistics*, 18(2):349–367.

Roberts, G. and Tweedie, R. (1996). Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363.

Roberts, G. O. and Rosenthal, J. S. (2007). Coupling and ergodicity of adaptive Markov chain Monte Carlo algorithms. *Journal of Applied Probability*, 44(2):458–475.

Roth, S. and Black, M. J. (2009). Fields of Experts. *International Journal of Computer Vision*, 82(2):205–229.

Rudin, L., Osher, S., and Fatemi, E. (1992). Nonlinear total variation based noise removal algorithms. *Physica D*, 60(1-4):259–268.

Rue, H. and Hurn, M. (1997). Loss functions for Bayesian image analysis. *The Art and Science of Bayesian Image Analysis*, pages 72–79.

Savage, J. and Chen, K. (2006). *On multigrids for solving a class of improved total variation based PDE models*, pages 69–94. Springer Berlin Heidelberg.

Scharstein, D. and Szeliski, R. (2003). High-accuracy stereo depth maps using structured light. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 195–202. Published by the IEEE Computer Society.

Schmidt, U., Gao, Q., and Roth, S. (2010). A generative perspective on MRFs in low-level vision. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1751–1758.

Tappen, M., Liu, C., Adelson, E., and WT (2007). Learning gaussian conditional random fields for low-level vision. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Thomas, D. B., Luk, W., Leong, P. H., and Villasenor, J. D. (2007). Gaussian random number generators. *ACM Computing Surveys*, 39(4):11–es.

Tikhonov, A. and Arsenin, V. (1977). *Solutions of ill-posed problems*. V. H. Winston and Sons.

van Meel, J. and Amolf, A. (2010). Molecular dynamics on a graphics card. ”http://www-old.amolf.nl/vanmeel/mdgpu/index.html; accessed 28-Sept-2010”.

Wang, Z., Bovik, A., Sheikh, H., and Simoncelli, E. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612.

Werlberger, M., Pock, T., and Bischof, H. (2010). Motion estimation with non-local total variation regularization. In *Proc. 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA*.

Zach, C., Pock, T., and Bischof, H. (2007). A Duality Based Approach for Realtime TV-L1 Optical Flow. *Proceedings of the 29th DAGM Symposium on Pattern Recognition*.

Zhu, M. and Chan, T. (2008). An efficient primal-dual hybrid gradient algorithm for total variation image restoration. *UCLA CAM Report*, (1):1–29.

Zhu, M., Wright, S. J., and Chan, T. F. (2008). Duality-based algorithms for total-variation-regularized image restoration. *Computational Optimization and Applications*.