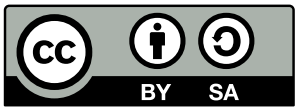
Saarland University
Faculty of Natural Sciences and Technology I
Department of Mathematics

Master's Thesis
Bregman Iteration for Optical Flow

submitted by
Laurent Hoeltgen

on August 25, 2010

Supervisor	Dr. Michael Breuß
Reviewer	Dr. Michael Breuß
Reviewer	Prof. Dr. Joachim Weickert



Statement

I hereby declare that this master's thesis has been written only by the undersigned and without any assistance from third parties.
Furthermore, I confirm that no sources have been used in the preparation of this thesis other than those indicated in the thesis itself.

Saarbrücken, August 25, 2010

Acknowledgements

I would like to express my sincere thanks to Dr. Michael Breuß for his excellent mentoring and support during the development of this thesis. Furthermore, I would also like to thank Henning Zimmer for his numerous suggestions and ideas on how to improve the optical flow models and algorithms used in this thesis.
Special thanks go to Prof. Dr. Joachim Weickert and Dr. Andrés Bruhn for sparking my interest in image processing and computer vision.
Last but not least, I have to express my special thanks to my friends and family, in particular Claudine Schiltz and Sylvain Delvaux, for their continuous support.

Abstract

Osher and his colleagues introduced Bregman iterations in image processing in 2005. This technique is known to yield excellent results for denoising/deblurring and compressed sensing tasks but it has so far been rarely used for other image processing problems. Some of the assets of the Bregman framework are the high flexibility and the existence of a thorough convergence theory. In this thesis we adapt the split Bregman iteration, originally developed by Goldstein and Osher, to the optical flow problem. The versatility of the Bregman framework allows us to present a general approach to solve variational formulations with modern data terms incorporating higher order constancy assumptions as well as discontinuity preserving smoothness terms such as the popular total variation regulariser. Several models will be analysed, and for each one a detailed algorithm based on the split Bregman iteration will be presented. Finally, we will analyse the theoretical properties of the Bregman iteration. The most interesting questions such as convergence and error estimation will be treated in detail, thus providing a solid mathematical basis for further research.

Contents

List of Figures

List of Tables

List of Algorithms

List of Symbols

$\mathrm{cl}\,(S)$	The closure of S.
$\mathrm{conv}\,(S)$	The convex hull of S.
$\mathrm{dom}\,f$	The effective domain of f.
$\mathrm{Epi}\,f$	The epigraph of f.
$\mathrm{int}\,(S)$	The interior of S.
$\mathcal{R}\,(A)$	Range of the operator A.
$\mathbb{N}$	$\{0, 1, 2, 3, \ldots\}$
$\mathbb{N}^*$	$\{1, 2, 3, \ldots\}$
$\|\cdot\|_\infty$	Operator norm.
$\overline{\mathbb{R}}$	The whole real line plus the symbols $+\infty$ and $-\infty$.
$\partial f\,(x)$	Subdifferential of f at x.
A^*	The adjoint operator of A.
$B_r\,(x)$	The open ball with radius r and center x.
$D_f^p\,(x, y)$	Bregman divergence of f at x and y with subgradient p.
$L_c\,(f)$	The lower levels set of f associated with c.
X^*	The dual space of X.

1 Introduction

The original work of Bregman

In 1967 L.M. Bregman presented an iterative algorithm to compute a common point of a family of convex sets [14]. The basic idea was to project a certain starting vector iteratively onto the sets of that family until it arrived in their intersection. Similar strategies were already known before, however the novelty in Bregman's approach was that he used non-orthogonal projections which allowed him to steer the iterates in such a way that the limiting point could even possess certain desired properties. These projections were computed with the help of a function that Bregman himself called the *D-projection* and which people later referred to as the *Bregman divergence* or *Bregman distance*. Bregman exploited the fact that he could steer the iterates and showed that it was possible to solve certain problems in convex programming. Unfortunately, his algorithm had a significant disadvantage. It could only be applied under rather strict conditions. The cost function of the convex programming problem for example had to be continuously differentiable and strictly convex. Nevertheless, his idea of using non-orthogonal projections in combination with this *D-projection* was interesting enough that people continued to research in that direction.

The Bregman iteration in image processing

In 2005 Stanley Osher and his colleagues proposed an algorithm for the iterative regularisation of inverse problems that was based on the Bregman divergence [62]. This algorithm, nowadays called *Bregman iteration*, possessed a high flexibility and was applicable under much weaker assumptions than Bregman's original formulation, thus rendering it particularly attractive for image processing tasks, where one often considers variational problems with non-differentiable/not strictly convex energies. Osher and his colleagues successfully used the algorithm for image restoration purposes such as denoising and deblurring. They were able to produce excellent results, especially in combination with the Rudin-Osher-Fatemi (ROF) model for denoising [71]. This made the Bregman iteration relatively popular, and many adaptations of the algorithm were published in the following years. Among the numerous application fields it has for example been used to solve the basis pursuit problem [28, 63, 94] and was subsequently applied to medical imaging problems in [47]. Further applications include deconvolution and sparse reconstruction [96], wavelet based denoising [92] and nonlinear inverse scale space methods [24, 26]. Generally speaking, the Bregman iteration can be applied to any problem that can be written under the form

$$\arg\min_{u} \; J(u) \quad \text{such that} \quad H(u) = 0 \tag{1.1}$$

with convex functions J and H and $H(u) \geqslant 0 \; \forall u$.

Algorithmic variants of the Bregman iteration

There exists also a certain number of variations of Osher's Bregman iteration. Among the most prominent ones are the split Bregman algorithm [43] and the linearised Bregman iteration [28].

The linearised Bregman formulation solves the same kind of problems as the standard Bregman iteration. The difference lies essentially within the fact that it approximates the constraining function H in eq. (1.1) by its first order Taylor expansion. This approach leads in certain cases, such as compressed sensing and sparse denoising (denoising of undersampled sparse signals) [28, 63], to extremely efficient algorithms with high convergence speeds.

On the other hand, the split Bregman algorithm is a formulation that solves a different kind of problems. It can be used to solve a rather large class of L_1 regularized problems in a fairly easy way, while offering, at the same time, good performance. It is capable of solving the following problems

$$\arg\min_u \|Au - b\|_1 + H(u) \tag{1.2}$$

with a matrix A, a vector b and a convex function H. The difficulty in solving such optimisation problems lies in the fact that the 1-norm is not differentiable. The trick behind the split Bregman algorithm is that one divides the initial problem into two subproblems that can be solved very efficiently. This is done by introducing an auxiliary variable d and considering the equivalent problem

$$\arg\min_u \|d\|_1 + H(u) \quad \text{such that} \quad \|d - Au - b\|_2^2 = 0 \tag{1.3}$$

By applying Osher's standard Bregman iteration and a few other transformations one arrives finally at an algorithm that consists essentially in solving the following two subproblems

$$\begin{aligned} &\arg\min_u \; H(u) + \|Au - d + b\|_2^2 \\ &\arg\min_d \|d\|_1 + \|d - Au - b\|_2^2 \end{aligned} \tag{1.4}$$

If for example $H(u) = \|Bu + c\|_2^2$ for some matrix B and vector c, then these two problems can be solved in a highly efficient manner since the first one can be reduced to solving a linear system and the second one possesses a well known analytical solution that can be formulated with the help of so called shrinkage operators.

The optical flow problem

The excellent results that have been achieved in the past with the Bregman framework motivate us to investigate if it can also be applied to another area in image processing, namely computer vision and there especially the optical flow problem. The optical flow problem is a highly ill-posed inverse problem. It consists in determining the displacement

(a) Frame 1 (b) Displacement field (c) Frame 2

Figure 1.1: Two consecutive images of the *Marbled-Block* sequence by Otte and Nagel [60] and the sought displacement field (author: Andrés Bruhn [19]).

field between different frames of a given image sequence. Figure 1.1 illustrates the problem: given two consecutive frames, one is interested in determining the displacement vector field that maps all points of the first frame onto their new location in the second frame. The difficulty lies essentially in the fact that in many cases this correspondence will not be unique or simply fails to exist because of various problems such as noise, illumination changes and overlapping objects.

From a practical point of view these problems allow the extraction of motion information such as the direction and velocity of moving objects. This knowledge is for example highly interesting in the context of object tracking, driver assistance systems, autonomous navigation of robots and video surveillance.

As for solving the optical flow problem, variational formulations and regularisation strategies belong to the most accurate and best understood techniques. They determine the unknown displacement field as the minimiser of an energy functional that penalises deviations from the model assumptions. If we denote by $\partial^k f$ the set of all partial derivatives of order k of an image sequence f, then the general form of a variational approach can be given by

$$\arg\min_{u} \int_{\Omega} D\left(\partial^k f, u\right) + S\left(\nabla f, \nabla u\right) \mathrm{d}x \tag{1.5}$$

where $D(\partial^k f, u)$ denotes a data term that represents one or more constancy assumptions on $\partial^k f$, while $S\left(\nabla f, \nabla u\right)$ stands for a smoothness term that assumes the flow field u to be smooth or piecewise smooth. The first such formulation was given by Horn and Schunck in their seminal work [50] and dates back to 1981. Variational formulations have the advantage that they allow a transparent modelling and that the resulting flow fields are dense and generally relatively accurate. However, the minimisation of the energy functional is usually highly non-trivial, especially for non-differentiable and/or non-convex formulations. Although numerous approaches have been proposed in the past to handle these problems, it is still a very active field of research. Highly efficient minimisation

schemes that would allow realtime performance are often only possible in combination with rather simple models such as the one of Horn and Schunck. On the other hand, more complex models, such as the complementary optic flow model of Zimmer et al. [98], yield much more accurate results but possess relatively high computation times.

Although many developments have been made on the modelling side (c.f. [9, 10, 18, 56, 58, 61, 85, 90, 93, 98] for a small account), there are just a few works concerned with the mathematical validation of the considered algorithms. In [53, 57] it has been shown that the classical approach of Horn and Schunck converges. Furthermore, the authors of [57] showed that the linear system obtained through the Euler-Lagrange equations has a symmetric and positive definite matrix and thus allows the usage of many efficient solvers. The authors of [86, 95] developed an algorithm that solves the so called TV-L^1 model through an alternating minimisation scheme. This algorithm gives excellent results in practice. However, it does not converge towards a solution of the orginal energy functional. The author of [19] discussed the usage of efficient algorithms such as the *Multigrid* approach [45] and the so called *Lagged-Diffusivity* or *Kačanov* method [31, 39, 54]. A certain number of convergence results can be found within this work and its references. Finally, it is also possible to consider the solutions of the Euler-Lagrange equations as a steady-state of a corresponding diffusion-reaction system that one may solve by means of a steepest descend approach [87, 88].

The Bregman iteration and the optical flow problem

In this thesis we present a novel method to solve the optical flow problem by combining the strengths of the Bregman framework with the advantages of variational formulations. The standard approach to solve variational formulations considers the Euler-Lagrange equations of the corresponding energy functional. This leads, in general, to a non-linear system of partial differential equations that is difficult to solve. Our approach will not make this detour to the Euler-Lagrange equations. Instead it minimises the energy functional directly by finding a suitable discretisation such that we can rewrite the optimisation problem in the form

$$\arg\min_{u,v} \ \left\|F\binom{u}{v} + f\right\|_2^2 + \lambda \left(\|\boldsymbol{\nabla} u + b_u\|_2 + \|\boldsymbol{\nabla} v + b_v\|_2\right), \quad \lambda > 0 \tag{1.6}$$

with matrices F and $\boldsymbol{\nabla}$ and vectors f, b_u and b_v. This unconstrained convex optimisation problem can then easily be solved with the split Bregman iteration. All in all, the final algorithm that we will obtain consists mostly in solving a large and extremely sparse linear system and solving an optimisation problem whose analytical solution is well known and which can easily be computed with shrinkage operators. The fact that our approach is based upon the split Bregman iteration also allows us to adapt the existing mathematically sound convergence theory of the Bregman framework to our optical flow algorithms. We will show in this context that the obtained iterates must necessarily converge towards a minimum of the energy functional. Further questions that are important for numerical purposes will also be discussed. We will for example show that the above mentioned linear system has a symmetric and positive definite matrix. Because

of the vast number of algorithms known to solve such systems, it will be possible to easily implement our new approach. This simplicity, when compared to the Euler-Lagrange formalism, certainly represents one of the assets of our new algorithm. Furthermore, our approach is highly flexible when it comes to the underlying model. It can be combined with functionals that use only quadratic terms as well as with formulations that use both quadratic and subquadratic terms or even with approaches that use subquadratic terms exclusively. As a consequence, simple energies such as the one used in the model of Horn and Schunck may just as well be used as more modern and complicated approaches such as the TV-L_1 model of Zach et al. [95].

Outline of the thesis

This thesis is organised as follows.

In Chapter 2 we will introduce the mathematical tools needed in the later chapters. Most results stem from functional and convex analysis. Important concepts are the notion of subgradients and the subdifferential calculus. The subgradient is a generalisation of the classical gradient and is a key element of the Bregman algorithm. Another interesting concept that we will present are the so called shrinkage operators. It is an intriguing fact that many unconstrained minimisation problems have an analytical solution that can be formulated via these operators.

In Chapter 3 we will present three variants of the Bregman algorithm and analyse their different properties. The first one will be the standard Bregman iteration developed by Osher and his colleagues in [62]. Then, we will present a slight modification of this algorithm that simplifies certain minimisation steps and thus yields a slightly better performance. Finally, we will consider the split Bregman algorithm from Goldstein and Osher as presented in [43]. We will also discuss the convergence behaviour of each approach and give conditions under which the algorithms are well defined. The fact that it is possible to present a solid theory of convergence for the Bregman iteration is one of the main reasons that make this algorithm so attractive.

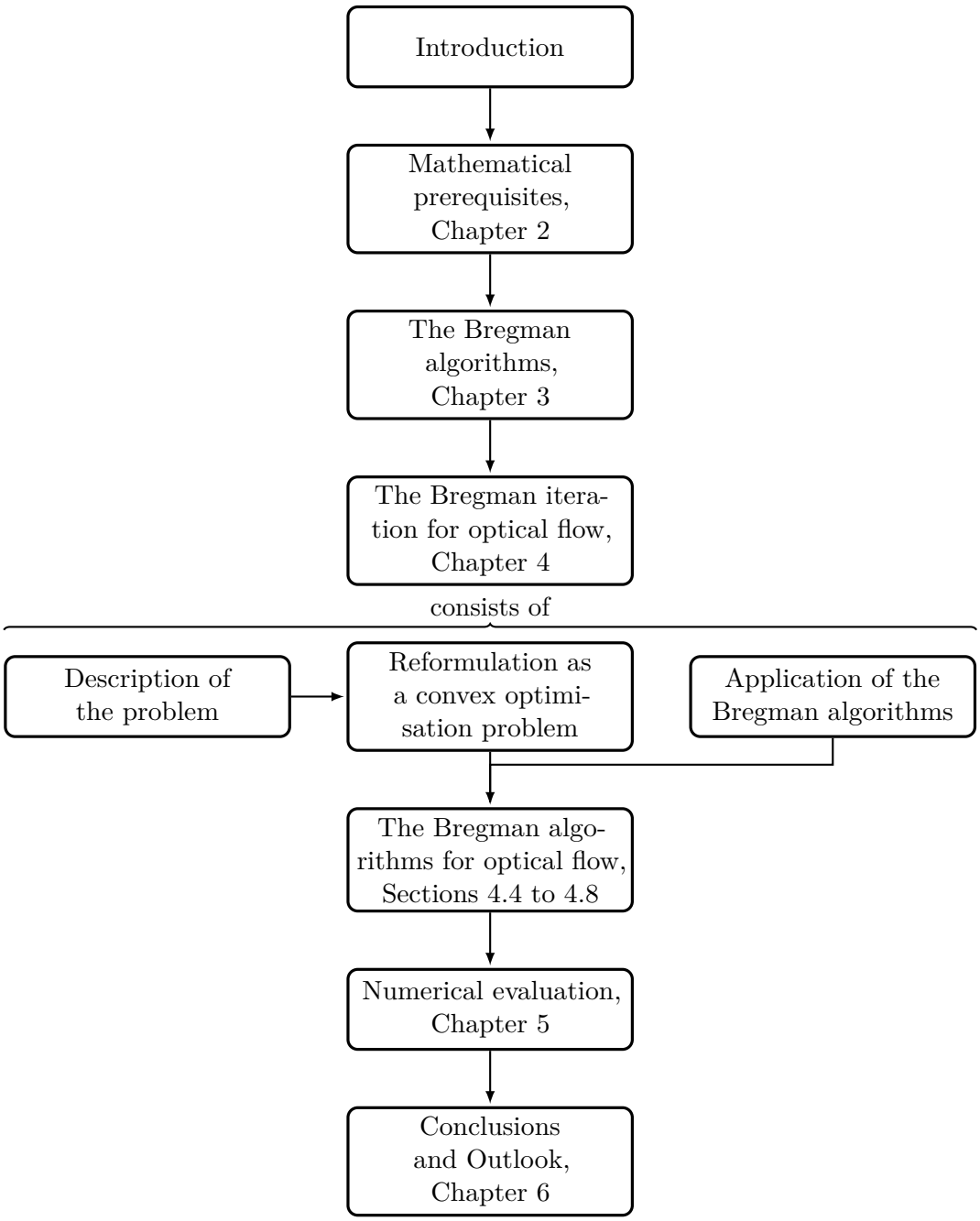
Chapter 4 will be dedicated to the optical flow problem. We will give a precise formulation of the problem and consider several variational models to compute the sought displacement field. We will then develop our algorithm by demonstrating that the Bregman framework can be used to compute the optical flow. A detailed formulation of the algorithm will be presented for each model that we consider and further ways to improve the formulations will be discussed. Chapter 4 together with Chapter 3 represent the main part of this thesis. They contain the core ideas as well as most of the important contributions.

In Chapter 5 we will prove empirically that the Bregman formulation is well suited for the optical flow problem by applying our algorithms against the test sequences of the Middlebury database [74]. Since the exact ground truths for these test sequences are known, we will be able to provide both quantitative and qualitative results.

Finally, in Chapter 6 we will conclude this thesis with a summary and an outlook on possible future research regarding the Bregman iteration.

This outline is also summarised in a more graphical form on the next page to allow a quicker reference.

Outline of the thesis



2 Mathematical prerequisites

In this chapter we will present the necessary mathematical tools that will allow us later on to perform a thorough analysis of the Bregman algorithms. Most of the following results stem from functional and convex analysis. We will assume that the reader is familiar with the notion of convex sets. Furthermore, we assume that the Hölder inequality and the operator norm are well known. The structure of this chapter is as follows: Section 2.1 will contain a certain number of preliminary results that we are going to need for the forthcoming tasks. Afterwards we will present a few of the appealing properties of convex sets and functions. One of this properties is that under certain rather weak conditions convex functions are continuous. This will be shown in Section 2.2. Finally, we will focus on the concept of subdifferentiability. Subdifferentials represent a generalisation of the classical gradient and are a powerful tool to compute the possible locations of extrema for convex functions. Furthermore, the Bregman framework makes intensive use of this notion and therefore, it will be necessary to study it in detail. All the important results will be shown in Sections 2.3 and 2.4.

Notations used throughout this chapter
In the following, X shall always denote a normed vector space over $\mathbb{R}$. Furthermore, X^* will be the dual space of X, i.e. the set of all linear and continuous maps from X to $\mathbb{R}$. For $x^* \in X^*$ and $x \in X$ we will denote $x^*(x)$ by $\langle x^*, x\rangle$. Finally, a convex function will usually map from X to $\mathbb{R}$. For further notations we refer to the list of symbols given on page xvii.

2.1 Fundamental results from functional analysis

The main goal of this section is to show that one can separate two disjoint convex sets by linear functionals. Separating two convex sets A and B means that there exists a continuous linear functional x^* and a number γ such that $\langle x^*, a\rangle < \gamma$ for all $a \in A$ and $\langle x^*, b\rangle \geqslant \gamma$ for all $b \in B$. We say that the separation is strict if the above mentioned inequalities hold with "$<$" and "$>$". Figure 2.1 depicts a graphical illustration of an example scenario in the two dimensional case. If the functional x^* is known explicitly, then it allows us to distinguish the elements from the sets A and B through a simple evaluation. In most cases however, we can only guarantee the existence of such a functional and know very little about its exact form. Fortunately, the knowledge of its existence is often sufficient, as we will see later in the proofs of Theorem 2.45 and Theorem 2.46. In Theorem 2.1 and Proposition 2.3 we are going to state the necessary conditions that allow such a separation. Theorem 2.1 is also known in the literature as the theorem of Hahn-Banach and it is one of the most important results from functional analysis.

We note that, except for Proposition 2.3, all the results in this section can also be found in [1, 2, 40, 70, 72, 83, 91]. Results such as the Theorem of Hahn-Banach are mostly presented to clarify which formulation we will be using, as there exists a certain number of variations with slightly different phrasings.

Theorem 2.1 (Theorem of Hahn-Banach: separation of convex sets)
Assume A, B are two convex, non-empty and disjoint subsets of X. If A is open, then there exists a $x^* \in X^*$ and $\gamma \in \mathbb{R}$, such that for all $a \in A$ and $b \in B$ we have

$$\langle x^*, a\rangle < \gamma \leqslant \langle x^*, b\rangle \tag{2.1}$$

If A is compact and B closed, then there exist $x^* \in X^*$, $\gamma \in \mathbb{R}$ and $\varepsilon > 0$ such that for all $a \in A$ and $b \in B$ we have

$$\langle x^*, a\rangle < \gamma - \varepsilon < \gamma < \langle x^*, b\rangle \tag{2.2}$$

Since both inequalities contain a "$<$", it follows that $x^* \not\equiv 0$.

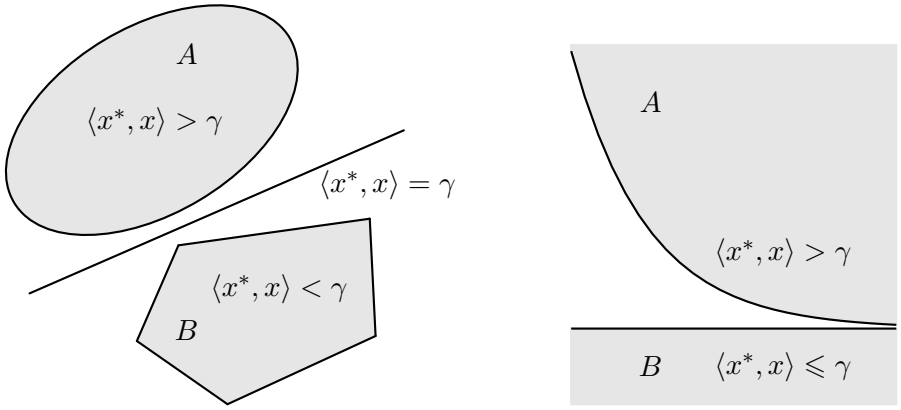
Proof: We will not proof this theorem here as it is a well known result from functional analysis. The complete proof would be rather lengthy and require the introduction of further concepts that would be of no use for us later on. Furthermore, the only point where we will need this theorem, will be to prove the correctness of Theorem 2.45 and Theorem 2.46. Therefore, we refer to [1] (Theorem 6.11, p. 61) or [72] (Theorem 3.4 p. 59) for a detailed proof. □

Note that the previous theorem only guarantees the existence of a functional x^*. It does not give us any information on its exact form.

As already mentioned, Fig. 2.1 is a graphical illustration of the Hahn-Banach theorem. The left picture presents a strict separation whereas the other picture illustrates a non-strict separation. The separating functional for the right picture would be $\langle x^*, (x, y)\rangle := y$ and $\gamma = 0$.

Theorem 2.1 states that one can separate two convex sets A and B if either A is open and B arbitrary or if A is compact and B closed. The theorem also requires that the sets have no common point. But what would happen if we relaxed the last requirement? What if we had two closed convex sets that only intersect at some boundary points? Theorem 2.1 clearly states that we can still separate the interior of one of the sets from the other. All one has to check is that $\operatorname{int}(A)$ remains convex if A is already convex. This assertion will be proven in Proposition 2.8. However, at this point it is unclear whether we can say something about the boundary. Such scenarios are going to appear in the proofs of Theorem 2.45 and Theorem 2.46 and therefore, they need to be discussed in detail. This will be done in Proposition 2.3. But before doing so, we need to introduce another necessary result which is also known in the literature as the line segment principle. It can be found completely with its proof in [70].

Proposition 2.2 (Line segment principle)
Assume $K \subseteq X$ is a convex set with $x_1 \in \operatorname{cl}(K)$ and $x_2 \in \operatorname{int}(K)$. Then we have $\lambda x_1 + (1-\lambda)\, x_2 \in \operatorname{int}(K)$ for any $\lambda \in [0, 1)$, i.e. the line segment $[x_2, x_1)$ is a subset of $\operatorname{int}(K)$.



(a) Strict separation by a linear functional x^*.

(b) Non-strict separation.

Figure 2.1: Separation of convex sets by a linear functional x^*. The second example uses the sets $A := \left\{(x,y) \in \mathbb{R}^2 \mid y > e^{-x}\right\}$ and $B := \left\{(x,y) \in \mathbb{R}^2 \mid y \leqslant 0\right\}$. The separating functional would be $\langle x^*, (x,y)\rangle := y$.

Proof: By assumption we have $x_2 \in \operatorname{int}(K)$ and thus there exists a $\varepsilon > 0$ with $B_\varepsilon(x_2) \subseteq K$. Now choose $y := \lambda x_1 + (1-\lambda)\, x_2$ for some $\lambda \in (0,1)$. We will show that $B_r(y) \subseteq K$ holds for $r := (1-\lambda)\,\varepsilon$. This would imply that $y \in \operatorname{int}(K)$. By choosing $z \in B_r(y)$ arbitrary, it follows that $\|y - z\| < r = (1-\lambda)\,\varepsilon$. On the other hand, $x_1 \in \operatorname{cl}(K)$ means that $B_\delta(x_1) \cap K \neq \emptyset$ for every $\delta > 0$. Especially for

$$\delta := \frac{(1-\lambda)\,\varepsilon - \|y - z\|}{\lambda} > 0 \tag{2.3}$$

there exists a $z_1 \in B_\delta(x_1) \cap K$, i.e. we have

$$z_1 \in K \text{ and } \|z_1 - x_1\| < \frac{(1-\lambda)\,\varepsilon - \|y - z\|}{\lambda} \tag{2.4}$$

The above described construction is also outlined in Fig. 2.2. Now we define

$$z_2 := \frac{1}{1-\lambda} z - \frac{\lambda}{1-\lambda} z_1 \tag{2.5}$$

For this z_2 we have

$$z_2 - x_2 = \frac{z - \lambda z_1}{1-\lambda} - x_2 = \frac{z - \lambda z_1 - (y - \lambda x_1)}{1-\lambda} \tag{2.6}$$

As a consequence we obtain

$$\begin{aligned} \|z_2 - x_2\| &\leqslant \frac{1}{1-\lambda}\left(\|z - y\| + \lambda\,\|x_1 - z_1\|\right) \\ &< \frac{1}{1-\lambda}\left(\|z - y\| + (1-\lambda)\,\varepsilon - \|z - y\|\right) \\ &= \varepsilon \end{aligned} \tag{2.7}$$

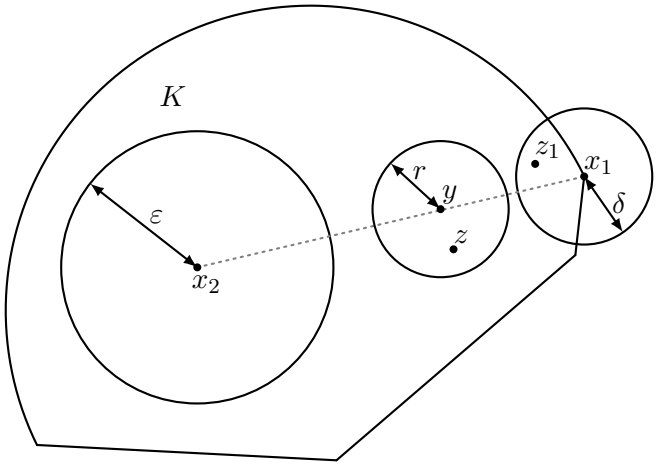


Figure 2.2: Construction used in the proof of Proposition 2.2.

Thus $z_2 \in B_\varepsilon(x_2)$ and $z_2 \in K$. Since $z = \lambda z_1 + (1-\lambda) z_2$ it follows from the convexity of K, that $z \in K$. But z was chosen arbitrarily from $B_r(y)$. Therefore, we must even have $B_r(y) \subseteq K$. □

Proposition 2.3
Assume that A and B are two convex and non-empty subsets of X with $\operatorname{int}(A) \neq \emptyset$ and $\operatorname{int}(A) \cap B = \emptyset$. Then there exists a $x^* \in X^*$ and $\gamma \in \mathbb{R}$, such that for all $a \in A$ and $b \in B$ we have

$$\langle x^*, a\rangle \leqslant \gamma \leqslant \langle x^*, b\rangle \tag{2.8}$$

and $\langle x^*, a\rangle < \gamma$ for all inner points of A.

Proof: Theorem 2.1 implies that there exist $x^* \in X^*$, $x^* \not\equiv 0$ and $\gamma \in \mathbb{R}$, such that

$$\langle x^*, a\rangle < \gamma \leqslant \langle x^*, b\rangle \tag{2.9}$$

for all points in $b \in B$ and inner points of $a \in A$. Now choose $a_1 \in \operatorname{int}(A)$ and $a_2 \in A$. It follows from Proposition 2.2, that $[a_1, a_2) \subset \operatorname{int}(A)$ and therefore, for every point $a_3 \in [a_1, a_2)$ we have $\langle x^*, a_3\rangle < \gamma$. By considering $a_3 \to a_2$ it follows $\langle x^*, a_2\rangle \leqslant \gamma$. □

As we can see from the previous proposition, the worst that can happen if two convex sets intersect at their boundary is that there exist elements $a \in A$ and $b \in B$ such that $\langle x^*, a\rangle = \gamma = \langle x^*, b\rangle$ holds. So we lose, in a certain sense, the ability to separate the sets A and B, because $\langle x^*, a\rangle = \gamma = \langle x^*, b\rangle$ does not necessarily mean that a and b are always elements of the intersection. The sets

$$A := \Big\{(x,y) \in \mathbb{R}^2 \;\Big|\; y \geqslant |x|\Big\} \quad \text{and} \quad B := \Big\{(x,y) \in \mathbb{R}^2 \;\Big|\; y \leqslant 0\Big\} \tag{2.10}$$

for example intersect at the origin and the separating function is given by $\langle x^*, (x,y)\rangle := y$ and $\gamma = 0$. Any point with y coordinate 0, fulfils $\langle x^*, (x,y)\rangle = 0$, but only the point

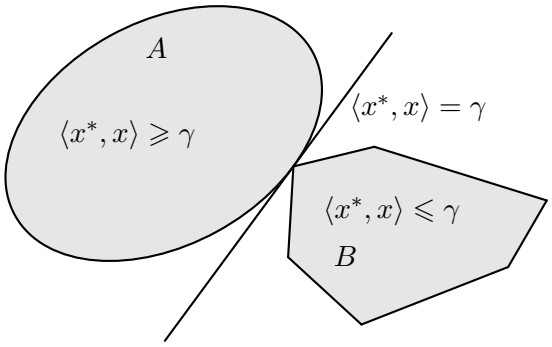


Figure 2.3: Example illustration of Proposition 2.3.

$(0,0)$ lies in A and B. All the others are only elements of B. The functional x^* cannot tell us whether such points belong to A or not.

However note that in general $\operatorname{int}(A)$ and B can still be separated, therefore, the loss is often not really substantial. In this context one may also keep Fig. 2.3 in mind. In this figure the polygon B touches the ellipse A on a boundary point. Proposition 2.3 says the we can still draw a line (in this case the tangent to the ellipse) that separates the interior of the ellipse from the polygon. Furthermore, the separation even holds for all but one boundary point of the ellipse, namely the point in the intersection.

We already mentioned that the separation results that we have seen until now will be used in the proof of the subdifferential sum formula (Theorem 2.45) in Section 2.3. The proof of this formula will require us to separate epigraphs of convex functions. An epigraph is a set in $X \times \mathbb{R}$ and therefore, we will have to work with functionals from $(X \times \mathbb{R})^*$. It is difficult to give a good description of the functionals from this space. However, the following proposition tells us that we can identify the space $(X \times \mathbb{R})^*$ with $X^* \times \mathbb{R}^* \cong X^* \times \mathbb{R}$. The latter space is much more comfortable to handle and will allow us to explain the underlying ideas of Theorem 2.45 in an intelligible manner. The here presented proof of this assertion is identical with the one found in [83].

Proposition 2.4 (Dual space of product spaces)
Assume X_1 and X_2 are two normed vector spaces over $\mathbb{R}$ and $1 < p, q < \infty$ such that $\frac{1}{p} + \frac{1}{q} = 1$. We define the spaces $X := X_1 \times X_2$ and $X' := X_1^* \times X_2^*$ equipped with the norms

$$\|(x_1, x_2)\|_X := \left(\|x_1\|_{X_1}^p + \|x_2\|_{X_2}^p\right)^{\frac{1}{p}}$$
$$\|(y_1, y_2)\|_{X'} := \left(\|y_1\|_{X_1^*}^q + \|y_2\|_{X_2^*}^q\right)^{\frac{1}{q}}$$

and consider the mapping $T : X' \to X^*$ defined by

$$\langle T(y), x\rangle := \langle y_1, x_1\rangle + \langle y_2, x_2\rangle \tag{2.11}$$

Then T is a linear isometric isomorphism between X' and X^*.

PROOF: The linearity of T is obvious. T is also continuous. In order to show this, assume that $y := (y_1, y_2) \in X'$ and define $x := (x_1, x_2) \in X$. Then it follows from the Hölder inequality

$$\begin{aligned} |\langle T(y), x\rangle| &\leqslant |\langle y_1, x_1\rangle| + |\langle y_2, x_2\rangle| \\ \text{Def. operator norm for } y_1,\, y_2 \to &\leqslant \|y_1\| \|x_1\| + \|y_2\| \|x_2\| \\ \text{Hölder inequality for sums} \to &\leqslant (\|x_1\|^p + \|x_2\|^p)^{\frac{1}{p}} (\|y_1\|^q + \|y_2\|^q)^{\frac{1}{q}} \\ &= \|x\|_X \|y\|_{X'} \end{aligned} \tag{2.12}$$

This implies the inequality $\|Ty\|_{X^*} \leqslant \|y\|_{X'}$ and thus $\|T\|_\infty \leqslant 1$. Now we show that $\|Ty\|_{X^*} = \|y\|_{X'}$ and thus $\|T\|_\infty = 1$ is true. For $y \equiv 0$ the statement is trivial. Therefore, assume now that $y \neq 0$. Consider $\varepsilon > 0$ and choose $z_k \in X_k$, $k = 1, 2$ such that $\|z_k\|_{X_k} = 1$ and $\|y_k\|_{X_k^*} \leqslant \langle y_k, z_k\rangle + \varepsilon$. Such z_k must necessarily exist because of the definition of the operator norm which is defined as the supremum over the set $\left\{|\langle y_k, z_k\rangle| \;:\; \|z_k\|_{X_k} = 1\right\}$. The supremum is the smallest upper bound, therefore, there exists for every $\varepsilon > 0$ a z_k with $\|z_k\|_{X_k} = 1$ such that $\|y_k\|_{X_k^*} - \varepsilon \leqslant |\langle y_k, z_k\rangle|$. If $\langle y_k, z_k\rangle \geqslant 0$, then we can omit the absolute value, otherwise simply consider $-z_k$. We define further $\tilde{x} := \left(\|y_1\|^{q-1} z_1, \|y_2\|^{q-1} z_2\right)$. From this we can conclude that

$$\begin{aligned} \|\tilde{x}\|_X^p &= \underbrace{\|z_1\|^p}_{=1} \|y_1\|^{p(q-1)} + \underbrace{\|z_2\|^p}_{=1} \|y_2\|^{p(q-1)} \\ &= \|y_1\|^{p(q-1)} + \|y_2\|^{p(q-1)} \\ &= \|y\|_{X'}^q \end{aligned} \tag{2.13}$$

where the relation $q = p(q-1)$ was used for the last equality. Using this estimation we can conclude that

$$\begin{aligned} \|y\|_{X'}^q &= \|y_1\|^q + \|y_2\|^q \\ &\leqslant \|y_1\|^{q-1} (\langle y_1, z_1\rangle + \varepsilon) + \|y_2\|^{q-1} (\langle y_2, z_2\rangle + \varepsilon) \\ \tilde{x}_k := \|y_k\|^{q-1} z_k \to &= \langle y_1, \tilde{x}_1\rangle + \langle y_2, \tilde{x}_2\rangle + \varepsilon \left(\|y_1\|^{q-1} + \|y_2\|^{q-1}\right) \\ &= \langle T(y), \tilde{x}\rangle + \varepsilon \left(\|y_1\|^{q-1} + \|y_2\|^{q-1}\right) \\ &\leqslant \|Ty\|_{X^*} \|\tilde{x}\|_X + \varepsilon \left(\|y_1\|^{q-1} + \|y_2\|^{q-1}\right) \\ &\leqslant \|Ty\|_{X^*} \|y\|_{X'}^{\frac{q}{p}} + \varepsilon \left(\|y_1\|^{q-1} + \|y_2\|^{q-1}\right) \end{aligned} \tag{2.14}$$

Multiplying both sides with $\|y\|_{X'}^{1-q} > 0$ gives (note: $\frac{q}{p} + 1 - q = 0$)

$$\|y\|_{X'} \leqslant \|Ty\|_{X^*} + \varepsilon \|y\|_{X'}^{1-q} \left(\|y_1\|^{q-1} + \|y_2\|^{q-1}\right) \xrightarrow{\varepsilon \to 0} \|Ty\|_{X^*} \tag{2.15}$$

Therefore, $\|Ty\|_{X^*} = \|y\|_{X'}$ and thus T is clearly injective. In order to show that T is also surjective, we choose $L \in X^*$ and $x_k \in X_k$, $k = 1, 2$ arbitrary. We define the

following two functionals $y_k \in X_k^*$, $k = 1, 2$ by

$$\begin{aligned} \langle y_1, x_1 \rangle &:= L(x_1, 0) \\ \langle y_2, x_2 \rangle &:= L(0, x_2) \end{aligned} \tag{2.16}$$

Then we have

$$\langle T(y), x \rangle = \langle y_1, x_1 \rangle + \langle y_2, x_2 \rangle = L(x_1, 0) + L(0, x_2) = L(x_1, x_2) \tag{2.17}$$

and thus $L = T(y)$. The continuity of the inverse T^{-1} follows immediately from the equality $\|Ty\|_{X^*} = \|y\|_{X'}$. This concludes the proof. □

We will now close this section with three results that have no direct link to the separation of convex sets but which will be used later on and actually belong to the domain of functional analysis.

In Section 2.3 we will introduce a generalisation of the concept of differentiability which is well suited for convex functions. The problem is that many convex functions have very good properties but fail to be differentiable. The euclidean norm in $\mathbb{R}^n$ is such an example. These generalised differentials will be called subdifferentials and one can show that many convex functions are subdifferentiable, i.e. they have a subdifferential at every point. Especially the subdifferentials of norms will be important for us later on. They will be presented in Proposition 2.48 and Corollary 2.49. The proofs of these results however will require that there exists in every normed vector space a continuous linear functional x_0^* and a vector x_0 such that

$$\langle x_0^*, x_0 \rangle = \|x_0\| \quad \text{and} \quad \|x_0^*\|_\infty = 1 \tag{2.18}$$

In order to guarantee the existence of such a linear functional, we need another variant of the Theorem of Hahn-Banach. Interestingly this version is frequently used to prove Theorem 2.1. The actual existence will be shown afterwards in Corollary 2.6.

Theorem 2.5 (Theorem of Hahn-Banach: extension of linear functionals)
Assume U a subspace of X equipped with the same norm as X. Then, for every linear functional $\tau \in U^*$ there exists a functional $T \in X^*$ such that the restriction of T onto U is equal to τ. Furthermore, we have $\|T\|_{X,\infty} = \|\tau\|_{U,\infty}$.

Proof: Again this is a well known result from functional analysis and will not be proven here since this theorem will only be used at a single occurrence and a complete proof would be rather lengthy. Therefore, we omit the demonstration and refer instead to [2] (Theorem 4.15 p. 181) or [91] (Theorem III.1.5 p. 97). □

The next corollary is closely related to the previous theorem. It can also be found with its proof within the just mentioned references.

Corollary 2.6
Assume X is a normed vector space. Then, for every $x_0 \neq 0$ there exists a functional $x_0^* \in X^*$ such that

$$\langle x_0^*, x_0 \rangle = \|x_0\| \quad \text{and} \quad \|x_0^*\|_\infty = 1 \tag{2.19}$$

PROOF: We consider $U := \operatorname{span}\{x_0\}$ and equip this subspace with the same norm as X. Then, every $y \in U$ is of the form $y = \alpha x_0$ with $\alpha \in \mathbb{R}$. We now define $\tau(y) := \alpha \|x_0\|$ on U. τ is a linear and continuous functional on U with norm 1 because all unit vectors in U simplify either to $\frac{x_0}{\|x_0\|}$ or to $\frac{-x_0}{\|x_0\|}$. Therefore, we have

$$\|\tau\|_\infty := \sup_{x,\|x\|=1} \|\langle \tau, x\rangle\| = \frac{\|x_0\|}{\|x_0\|} = 1 \tag{2.20}$$

The statement follows now immediately from Theorem 2.5. □

Definition 2.7 (Adjoint Operator)
Let X, Y be two normed vector spaces and A a continuous linear operator from X to Y. Then we define the *adjoint operator* $A^* : Y^* \to X^*$ of A by

$$\langle A^* y^*, x\rangle := \langle y^*, Ax\rangle \quad \forall y^* \in Y^*,\ x \in X \tag{2.21}$$

From this definition we can conclude immediately that the adjoint operator is uniquely defined and linear. If A is a matrix, then it follows from the definition that the adjoint operator is simply the transpose of the matrix A. We will encounter the adjoint operator again in Theorem 2.46. This theorem will be of great importance for the Bregman framework as it will reduce significantly the complexity of the algorithms presented in Chapter 3.

2.2 Elementary notions about convex functions and sets

The goal of this section will be to recall some elementary notions about convex functions and to prove a certain number of useful properties. The ultimate goal will be to show under which conditions convex functions are continuous. This result will be presented in Theorem 2.30. Similarly as for the previous section all the results presented here can easily be found in the literature. The following references were used: [17, 40, 49, 51, 52, 68, 69, 70]. The first half of this section (everything until Proposition 2.17) is mostly based on the presentation given in [17, 68, 69], whereas the second half is based almost exclusively on the results as stated in [68].

We will call a function *convex* if it fulfils the inequality

$$f(\lambda x + (1-\lambda) y) \leqslant \lambda f(x) + (1-\lambda) f(y) \tag{2.22}$$

for all x, y in X and $\lambda \in [0,1]$. If this inequality is strict for $x \neq y$, then we speak of *strict convexity*.

A certain number of statements in this section will be local statements, i.e. they tell us something about the properties of a function in a neighbourhood of an arbitrary point. The following simple observation from [68] shows, that all these cases can be reduced to a single one which is, in general, more comfortable to handle. Assume that f is convex on an open set U and that $x_0 \in U$. Now define

$$V := \{x \in X \mid (x + x_0) \in U\} \tag{2.23}$$

$$g(x) := f(x + x_0) - f(x_0) \tag{2.24}$$

Then it is clear that V is an open set around 0 and that $g(0) = 0$. Furthermore, the functions g and f share all important characteristics such as continuity, differentiability, convexity, etc. Therefore, one can usually make the simplifying assumption that $x_0 = 0$ and that $f(x_0) = 0$ when one has to prove local characteristics.

We will now present a rather general overview of the properties of convex functions. Special attention will be paid to their relationship with convex sets and epigraphs, as this will be important for us in the proofs of Theorem 2.45 and Theorem 2.46. Proposition 2.12, Proposition 2.15 and Proposition 2.16 are especially important in this context. Another point of interest will be the existence of global minima of convex functions. This will be discussed in Proposition 2.17. The reason why this result is so important, is that later on in Chapter 3 and Chapter 4 we are going to solve convex optimisation problems where it is essential to know that minimisers exist. The first proposition that we are going to state now has no direct link with convex functions. We claimed in the previous section that the interior of a convex set is also convex, but did not prove this statement at that time. Therefore, the proof is given in all its details now. The proof stems from [70] where this result was discussed within the context of variational analysis.

Proposition 2.8
Assume $K \subset X$ is a convex set with non-empty interior. Then its interior $\operatorname{int}(K)$ and its closure $\operatorname{cl}(K)$ are also convex.

PROOF: Choose $x, y \in \operatorname{int}(K)$ and consider $z = \lambda x + (1-\lambda) y$ with $\lambda \in (0,1)$. For any $u \in X$ we actually have $z + u = \lambda\left(x + \frac{u}{\lambda}\right) + (1-\lambda) y$. Now, if $\|u\|$ is sufficiently small, then it follows that $x + \frac{u}{\lambda} \in K$ and consequently we must have $z + u \in K$ since we assumed K to be convex. This again implies that $z \in \operatorname{int}(K)$ and therefore, we can conclude that the interior $\operatorname{int}(K)$ must be a convex set as well.

Next let $x, y \in \operatorname{cl}(K)$. Then there exist sequences $\{x_i\}_i$ and $\{y_i\}_i$ in K converging to x and y respectively. For any $\lambda \in (0,1)$ the sequence $\lambda x_i + (1-\lambda) y_i$ converges to $\lambda x + (1-\lambda) y$, so $\operatorname{cl}(K)$ must also be convex. □

Definition 2.9 (Effective domain of a convex function)
We define the *effective domain* of a convex function as the set of all x such that $f(x)$ is finite:

$$\operatorname{dom} f := \{x \in X \mid -\infty < f(x) < \infty\} \tag{2.25}$$

It is a prevailing custom in convex analysis not to restrict oneself to the class of all convex functions with a common fixed effective domain, but rather to consider functions defined on the whole space X (see for example [17, 69]). These functions may then of course also attain the values $\pm\infty$. This approach has the advantage that technical nuisances about the effective domain usually can be suppressed almost entirely. For example, when a convex function is constructed according to certain formulas, the same formulas specify the effective domain of f implicitly, because they specify where $f(x)$ is or is not $+\infty$. Without this approach one would always have to describe the effective domain explicitly before the values of f on that domain could be given. However, in this setting we might very well encounter arithmetic computations that include $\pm\infty$. Thus one has to be

careful to avoid undefined forms such as $\infty - \infty$. We will adopt the usual convention (as described in [69]) for computations with $\pm\infty$:

- $\alpha + \infty = +\infty + \alpha = +\infty$ for $-\infty < \alpha \leqslant +\infty$
- $\alpha - \infty = -\infty + \alpha = -\infty$ for $-\infty \leqslant \alpha < +\infty$
- $\alpha\,(\pm\infty) = \pm\infty\alpha = \pm\infty$ for $0 < \alpha \leqslant +\infty$
- $\alpha\,(\pm\infty) = \pm\infty\alpha = \mp\infty$ for $-\infty \leqslant \alpha < 0$
- $0\,(\pm\infty) = \pm\infty 0 = 0$
- $-\,(-\infty) = +\infty$

The reason for the above described approach lies in the fact that convex functions, like the Fenchel conjugate of a function $f : \mathbb{R}^n \to \mathbb{R}$, may attain the value $+\infty$. The Fenchel conjugate of a convex function f is defined by

$$f^*\,(\bar{x}) := \sup_{x \in \mathbb{R}^n} \{\langle \bar{x}, x\rangle - f\,(x)\} \tag{2.26}$$

and is used for example to show the existence of solutions to certain convex optimisation problems. It is easy to see that if $f \equiv 0$ then f^* is 0 in 0 and $+\infty$ anywhere else. We refer to [69] for more information about the Fenchel conjugate.

In this context certain authors also speak of *proper convex functions*. A proper convex function satisfies $f\,(x) < +\infty$ for at least one x and $f\,(x) > -\infty$ for all x. Proper convex functions are an effective way to avoid undefined forms involving ∞. We will from now on always assume that our convex functions are actually proper convex functions. Further, $\overline{\mathbb{R}}$ will denote in the following the whole real line plus the symbols $+\infty$ and $-\infty$.

Definition 2.10 (Level set)
Given a scalar $c \in \mathbb{R}$ and a function $f : X \to \overline{\mathbb{R}}$, the (lower) *level set* of f associated with c is given by

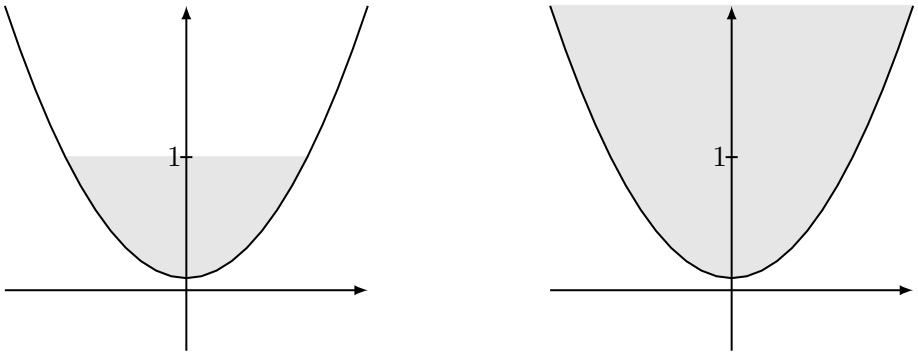
$$L_c\,(f) := \{x \in X \mid f\,(x) \leqslant c\} \tag{2.27}$$

Definition 2.11 (Epigraph)
The *epigraph* of a function $f : X \to \overline{\mathbb{R}}$ is defined as

$$\operatorname{Epi} f := \{(x, \alpha) \in X \times \mathbb{R} \mid f\,(x) \leqslant \alpha\} \tag{2.28}$$

From a geometric point of view, the epigraph of a function from $\mathbb{R}$ to $\mathbb{R}$ is the set of all points that lie above the graph of the function f. Figure 2.4 visualises the difference between a level set and an epigraph.

The next proposition illustrates the fundamental link between convex functions and convex sets. For every convex function from X to $\overline{\mathbb{R}}$, there exists a corresponding convex set in $X \times \mathbb{R}$. This relationship makes it easy to pass back and forth between a geometric and analytic approach in the analysis of convex functions. The proof presented here is a generalisation of the proof found in [52]. There, the author restricted himself to convex functions from $\mathbb{R}^n$ to $\mathbb{R}$.



(a) Level set of f associated with 1. (b) Epigraph of the same function f.

Figure 2.4: Illustration of a level set and an epigraph (marked in grey).

Proposition 2.12
A function $f : X \to \overline{\mathbb{R}}$ is convex if and only if Epi f is convex.

Proof: Assume that f is convex and that (x, α), (y, β) are elements of the epigraph. Then it follows that we have for all $\mu \in [0, 1]$

$$f(\mu x + (1-\mu) y) \leqslant \mu f(x) + (1-\mu) f(y) \leqslant \mu\alpha + (1-\mu)\beta \tag{2.29}$$

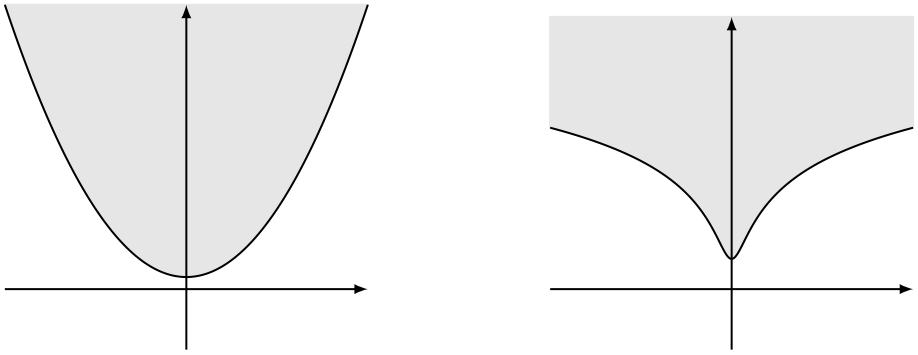
and therefore, $\mu(x, \alpha) + (1-\mu)(y, \beta) \in \text{Epi}\, f$. But this implies that Epi f is convex.

On the other hand, if Epi f is convex, then clearly $(x, f(x))$ and $(y, f(y))$ are elements of the epigraph whenever x and y lie in the effective domain of f and it follows also, that for all $\mu \in [0, 1]$ the point $\mu(x, f(x)) + (1-\mu)(y, f(y))$ is an element of the epigraph as well. Therefore, by the definition of Epi f, we have

$$f(\mu x + (1-\mu) y) \leqslant \mu f(x) + (1-\mu) f(y) \tag{2.30}$$

If x or y is not an element of dom f, then the above equation is trivially fulfilled. Thus f is convex. □

As we can see, the epigraph is of high importance in the study of convex functions. We deem it necessary to present at least some of its properties and to discuss their consequences. These results will be given in the next few statements. They show, that to a certain extend, the continuity of a convex function is linked to topological properties of its epigraph, such as closedness and non-empty interior. These observations will be crucial in the proof of Theorem 2.45. Proposition 2.13, stated hereafter, was already treated in [69]. We will present a slightly more general formulation by not restricting ourselves to $\mathbb{R}^n$. However, the proof remains identical to the one found in [69]. Proposition 2.16 originates from [17], although it was stated there without a proof. We will prove it here with a short and elegant argumentation. The ideas for our reasoning stem from observations found in [37], where the authors intended to solve variational problems and discussed the local boundedness of convex functions in a much more general setting than we will need it here. Finally Proposition 2.15 can be found almost verbatim in [70] and Proposition 2.17 is presented in a similar formulation in [52].



(a) A convex function with a convex epigraph.

(b) A non-convex function with a non-convex epigraph.

Figure 2.5: Graphical illustration of Proposition 2.12.

Proposition 2.13
If $f : X \to \overline{\mathbb{R}}$ is a convex function, then its effective domain is a convex subset of X.

Proof: Assume $x, y \in \operatorname{dom} f$. Then clearly $(x, f(x))$ and $(y, f(y))$ are elements of the epigraph. Since the epigraph of a convex function is also convex, it follows that

$$(\lambda x + (1-\lambda) y,\ \lambda f(x) + (1-\lambda) f(y)) \in \operatorname{Epi} f \quad \forall \lambda \in [0,1] \tag{2.31}$$

From this it follows immediately that

$$|f(\lambda x + (1-\lambda) y)| < \infty \tag{2.32}$$

and therefore, we have $\lambda x + (1-\lambda) y \in \operatorname{dom} f$. But this means that the effective domain of f is convex. □

Definition 2.14 (Lower semi-continuous)
A function f is *lower semi-continuous* at a point x_0 if for every sequence $(x_k)_k$ converging to x_0 we have

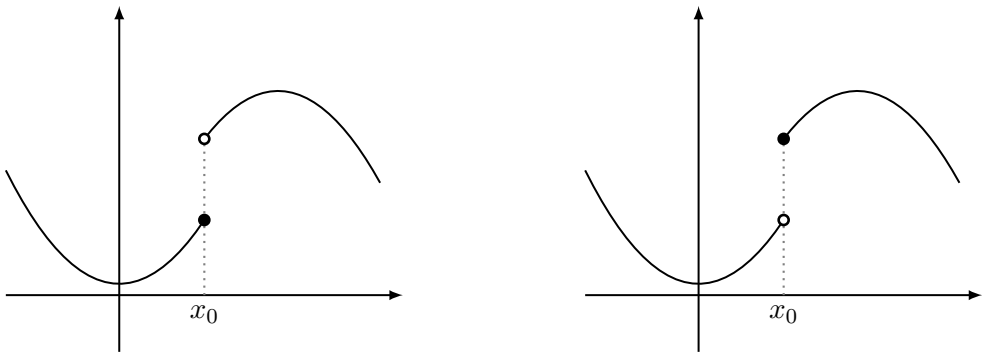
$$f(x_0) \leqslant \liminf_{k\to\infty} f(x_k) \tag{2.33}$$

We say that f is lower semi-continuous over a set $S \subseteq X$, if f is lower semi-continuous at every $x \in S$.

Figure 2.6 illustrates the idea behind lower semi-continuity. The left picture depicts a lower semi-continuous function. The function in the right picture is not lower semi-continuous in x_0. If one chooses a sequence $\{x_k\}_k$ approaching x_0 from the left, then the limit inferior will be strictly smaller than $f(x_0)$. Also note that every continuous function is obviously lower semi-continuous.

Proposition 2.15
For a function $f : X \to \mathbb{R} \cup \{+\infty\}$ the following three properties are equivalent:

1. f is lower semi-continuous over X



(a) Lower semi-continuous in x_0.

(b) Not lower semi-continuous in x_0.

Figure 2.6: Illustration of lower semi-continuity. The solid dot indicates $f(x_0)$.

2. $\operatorname{Epi} f$ is a closed set in $X \times \mathbb{R}$

3. The level sets $L_r(f)$ are closed (possibly empty) for all $r \in \mathbb{R}$.

PROOF: 1 $\Rightarrow$ 2: Let $(y_k, r_k)_k$ be a sequence of $\operatorname{Epi} f$ converging to (x, r) for $k \to \infty$. Since $f(y_k) \leqslant r_k$ for all k, the lower semi-continuity of f readily gives

$$r = \lim_k r_k \geqslant \liminf_k f(y_k) \geqslant \liminf_{y \to x} f(y) \geqslant f(x) \tag{2.34}$$

Implying that the epigraph must be closed.

2 $\Rightarrow$ 3: If $\operatorname{Epi} f$ is closed, then the set

$$F_r := \{(x, r) \mid x \in X,\ f(x) \leqslant r\} = \operatorname{Epi} f \cap (X \times \{r\}) \tag{2.35}$$

is closed in $X \times \mathbb{R}$. Now consider the embedding $j_r : X \to X \times \mathbb{R}$ defined by $x \mapsto (x, r)$. Because of the continuity of j_r it follows that $L_r(f) = j_r^{-1}(F_r)$ must also be closed.

3 $\Rightarrow$ 1: Suppose that f is not lower semi-continuous at some x: then, there is a sequence $(y_k)_k$ converging to x such that $f(y_k)$ converges to $\rho < f(x)$. Now pick any $r \in (\rho, f(x))$: for k large enough we have $f(y_k) \leqslant r < f(x)$; hence $L_r(f)$ contains the tail of $(y_k)_k$ but not its limit x. Consequently, this $L_r(f)$ is not closed. □

Proposition 2.16
Assume f is a convex function and continuous in $x \in \operatorname{int}(\operatorname{dom} f)$, then $\operatorname{int}(\operatorname{Epi} f)$ is a non-empty set.

PROOF: Since $x \in \operatorname{int}(\operatorname{dom} f)$ and f is continuous in x, there exists an open neighbourhood O of x in $\operatorname{dom} f$, such that f is bounded by a constant c on O. Then $O \times (c, +\infty)$ is an open subset of $X \times \mathbb{R}$. Furthermore,

$$O \times (c, +\infty) \subseteq \operatorname{Epi} f \tag{2.36}$$

It follows that for any $\alpha > c$, the point (x, α) is an inner point of $\operatorname{Epi} f$. □

The next proposition gives us criteria for global minimisers of convex functions. It is of fundamental importance for numerical considerations as it implies that minimisation algorithms applied on convex functions cannot get stuck in local minima.

Proposition 2.17
Assume that f is convex and that $\operatorname{dom} f$ is an open set. Then every local minimum of f in $\operatorname{dom} f$ is already a global minimum. Furthermore, if f is even strictly convex, then the minimum (in case it exists) is unique.

PROOF: Assume x is a local minimum and U a small open neighbourhood of x in $\operatorname{dom} f$ with radius ε. Then $f(x) \leqslant f(z)$ for all $z \in U$. Furthermore, assume that there exists $y \in \operatorname{dom} f$ such that $f(y) < f(x)$. Because of the convexity of f it follows that for every $\lambda \in (0,1)$ we have

$$\begin{aligned} f(\lambda y + (1-\lambda)x) &\leqslant \lambda f(y) + (1-\lambda) f(x) \\ &< \lambda f(x) + (1-\lambda) f(x) \\ &= f(x) \end{aligned} \tag{2.37}$$

This is a contradiction, since for sufficiently small λ we have

$$\|x - (\lambda y + (1-\lambda)x)\| = \lambda \|x - y\| < \varepsilon \tag{2.38}$$

and thus $\lambda y + (1-\lambda)x$ would be an element of U. Therefore, every local minimum is also a global minimum.

Now assume that f is strictly convex and that x and y are two different global minima. Then

$$f\left(\frac{1}{2}x + \frac{1}{2}y\right) < \frac{1}{2}f(x) + \frac{1}{2}f(y) = f(x) = f(y) \tag{2.39}$$

which is impossible, since x and y are global minima. Thus, the global minimum must be unique for strictly convex f. □

Remark 2.18
Note that the previous proposition does not imply that a minimum must always exist. The function could be unbounded or the infimum could simply not be attained in a finite point. One of the simplest examples is the exponential function. It is strictly convex but reaches is minimum for $x \to -\infty$.

Propositions 2.19, 2.20 and 2.22 and Definitions 2.21 and 2.23, that we will present now, are preliminary results that we will need for the proof of the main goal of this chapter, namely the continuity of convex functions. The formulations given here can also be found almost verbatim in [17, 68]. The continuity of convex functions will be discussed afterwards in Theorem 2.30.

Proposition 2.19
Assume $f : \mathbb{R} \to \mathbb{R}$ is convex and $x, y \in \mathbb{R}$. Then for $\lambda \in (0,1)$ and $z = \lambda x + (1-\lambda)y$ we have

$$\frac{f(z) - f(x)}{z - x} \leqslant \frac{f(y) - f(x)}{y - x} \leqslant \frac{f(y) - f(z)}{y - z} \tag{2.40}$$

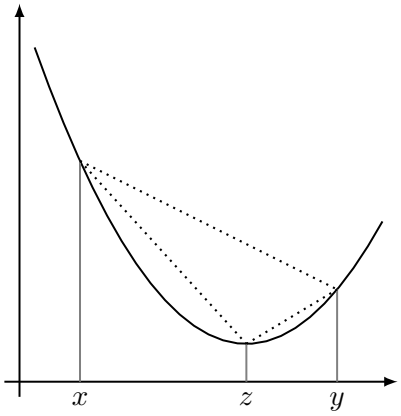


Figure 2.7: Visualisation of Proposition 2.19.

Geometrically this means that the slope of the line that connects the points $(x, f(x))$ and $(z, f(z))$ is smaller than the slope of the line that connects the points $(x, f(x))$ and $(y, f(y))$. The latter is itself again smaller than the slope of the line that connects the points $(z, f(z))$ and $(y, f(y))$. A graphical representation is shown in Fig. 2.7.

PROOF: This is a matter of simple computations. The first inequality follows from

$$\begin{aligned} \frac{f(z) - f(x)}{z - x} &= \frac{f(\lambda x + (1-\lambda) y) - f(x)}{\lambda x + (1-\lambda) y - x} \\ &\leqslant \frac{\lambda f(x) + (1-\lambda) f(y) - f(x)}{\lambda x + (1-\lambda) y - x} \\ &= \frac{f(y) - f(x)}{y - x} \end{aligned} \tag{2.41}$$

And the second is easily deduced from

$$\begin{aligned} \frac{f(y) - f(z)}{y - z} &= \frac{f(y) - f(x + (1-\lambda) y)}{y - \lambda x - (1-\lambda) y} \\ &\geqslant \frac{f(y) - \lambda f(x) - (1-\lambda) f(y)}{y - \lambda x - (1-\lambda) y} \\ &= \frac{f(y) - f(x)}{y - x} \end{aligned} \tag{2.42}$$

□

Proposition 2.20
A set K is convex if and only if for every $n \in \mathbb{N}$, the convex combination $\sum_{i=1}^{n} \lambda_i x_i$, with $x_i \in K$, $\lambda_i > 0$, $i = 1, \ldots, n$ and $\sum_{i=1}^{n} \lambda_i = 1$, is an element of K.

PROOF: If any convex combination of elements from K is again in K, then K is obviously convex. It is enough to consider $n = 2$.

Conversely, assume that K is convex. We will prove the statement by induction. For $n = 2$, we simply have the definition of the convexity of a set. Therefore, we assume now that $n > 2$ and that the assertion holds for $n - 1$. Consider the convex combination

$$x = \sum_{i=1}^{n} \lambda_i x_i, \quad x_i \in K, \quad \sum_{i=1}^{n} \lambda_i = 1 \tag{2.43}$$

where $\lambda_i < 1$ for all i. We have to show that this x is an element of K. We rewrite x under the following form

$$x = \lambda_n x_n + (1 - \lambda_n)\, y, \quad y = \sum_{i=1}^{n-1} \frac{\lambda_i}{1 - \lambda_n} x_i \tag{2.44}$$

Since $\sum_{i=1}^{n-1} \frac{\lambda_i}{1-\lambda_n} = 1$ it follows from the induction hypothesis that $y \in K$ and from this that $x \in K$ as well. □

Definition 2.21 (Convex hull)
Assume $S \subset X$. Then we call the set

$$\operatorname{conv}(S) := \bigcap_{\substack{S \subseteq K \subseteq X \\ K \text{ convex}}} K \tag{2.45}$$

the *convex hull* of S in X. Since intersections of convex sets are again convex, it follows that the convex hull of a set S is the smallest convex set containing S.

Proposition 2.22
Assume $S \subseteq X$. Then

$$\operatorname{conv}(S) = \{x \in X \mid x \text{ is a convex combination of elements from } S\} \tag{2.46}$$

Proof: We denote the righthand side of eq. (2.46) by C. If $K \supseteq S$ is a convex set, then it follows from Proposition 2.20 that $K \supseteq C$ and therefore, $\operatorname{conv}(S) \supseteq C$. Obviously we also have $S \subseteq C$. Therefore, it will now be enough to show that C is convex, because this implies that $\operatorname{conv}(S) \subseteq C$ must also hold. We consider $x, y \in C$ with

$$x = \sum_{i=1}^{n} \lambda_i x_i, \quad y = \sum_{j=1}^{m} \mu_j y_j \tag{2.47}$$

For $\alpha \in [0, 1]$ we have

$$\alpha x + (1 - \alpha)\, y = \sum_{i=1}^{n} \alpha \lambda_i x_i + \sum_{j=1}^{m} (1 - \alpha)\, \mu_j y_j \in C \tag{2.48}$$

since the coefficients fulfil the requirements of a convex combination. □

Definition 2.23 (Simplex)
Assume that the $n+1$ vectors x_i, $i = 0, \ldots, n$ are affine independent (i.e. the n vectors $x_0 - x_i$, $i = 1, \ldots, n$ are linearly independent). Then we call $\operatorname{conv}(\{x_0, \ldots, x_n\})$ a (n-dimensional) *simplex*. The elements x_i are called the corners of the simplex.

Remark 2.24
From Proposition 2.22 it follows immediately, that we can write every element of the simplex $\operatorname{conv}(\{x_0, \ldots, x_n\})$ as a convex combination of the corners of this simplex

$$x = \sum_{i=0}^{n} \lambda_i x_i$$

At this point we have completed all the necessary preparations to show the main result of this chapter, namely that convex functions on $\mathbb{R}^n$ are continuous. This statement will be proven in several steps. Starting with a local boundedness assumption, we will improve the properties of our convex function step by step until we arrive at the continuity. The proof that we will present actually holds for any convex function that is locally bounded, but in general a convex function need not to have this characteristic. In $\mathbb{R}^n$ however, convex functions are always locally bounded. This will be proven with the help of the simplexes from Definition 2.23. The results formulated in Proposition 2.25 and Theorems 2.26 and 2.28 to 2.30 all originate from [68]. We will present a slight generalisation of these statements. The authors of [68] restricted themselves to convex functions that are finite everywhere. We will show that this restriction is not necessary.

Proposition 2.25
Let f be a convex function on X. If f is bounded above in an ε-neighbourhood of some point x_0, then it is bounded from below in the same neighbourhood.

Proof: Without loss of generality, we can assume that this point is $x_0 = 0$ and suppose that f is bounded above by B in a ε-neighbourhood N_ε of x_0. Since

$$0 = \frac{1}{2}x + \frac{1}{2}(-x), \; f(0) \leqslant \frac{1}{2}f(x) + \frac{1}{2}f(-x) \tag{2.49}$$

we have $f(x) \geqslant 2f(0) - f(-x)$. Now $\|x\| \leqslant \varepsilon$ implies $\|-x\| \leqslant \varepsilon$ so that $f(-x) \leqslant B$ and

$$f(x) \geqslant 2f(0) - B \tag{2.50}$$

meaning that f is bounded from below. □

Theorem 2.26
Let f be convex and $U \subseteq \operatorname{dom} f$ an open set. If f is bounded from above in a neighbourhood of $x_0 \in U$, then it is locally bounded. That means each $x \in U$ has a neighbourhood on which f is bounded.

Proof: We can safely assume that $x_0 = 0$. By Proposition 2.25 it follows that f is bounded on an ε-neighbourhood N of 0 by some B. Now take $y \in U$, $y \neq 0$ and $\rho > 1$ so that $z = \rho y \in U$ still holds. Define $\lambda = \frac{1}{\rho}$ and consider

$$M := \{v \in X \mid v = (1-\lambda)x + \lambda z, \; x \in N\} \tag{2.51}$$

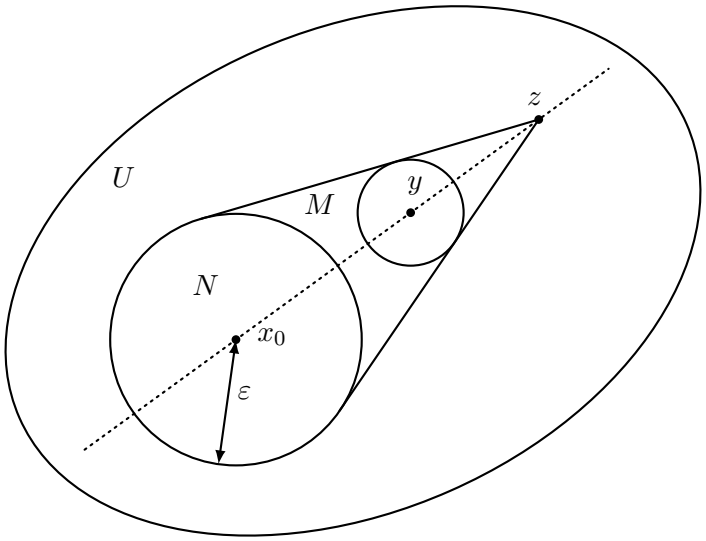


Figure 2.8: Construction described in Theorem 2.26.

Figure 2.8 shows a simple sketch of the sets U, N and M. For $x = 0$, we have $v = \lambda z = y$. Furthermore, if $\|x - 0\| = \|x - x_0\| = \varepsilon$, then

$$\|v - y\| = \|(1 - \lambda)\, x + \lambda z - \lambda z\| = (1 - \lambda)\, \varepsilon \tag{2.52}$$

Thus M is a neighbourhood of $y = \lambda z$ with a radius of $(1 - \lambda)\, \varepsilon$. Moreover the convexity of f implies that for every $v \in M$ we have

$$f(v) \leqslant (1 - \lambda)\, f(x) + \lambda f(z) \leqslant B + f(z) < \infty \tag{2.53}$$

and as a consequence, f is bounded above on M. By Proposition 2.25, it follows that f is also bounded from below on M. □

Theorem 2.26 is highly interesting. It says that local boundedness in a single point is enough for a convex function to be locally bounded in every point. This is by no means a trivial statement.

Definition 2.27 (Locally Lipschitz function)
A function f defined on an open set $U \subseteq X$ is said to be *locally Lipschitz*, if for each $x \in U$ there exists a ε-neighbourhood $N_\varepsilon(x)$ and a constant $K(x)$ such that, if $y, z \in N_\varepsilon(x)$, then

$$|f(y) - f(z)| \leqslant K \|y - z\| \tag{2.54}$$

If this inequality holds throughout a set $V \subseteq U$ with K independent of x, then we say that f is Lipschitz on V.

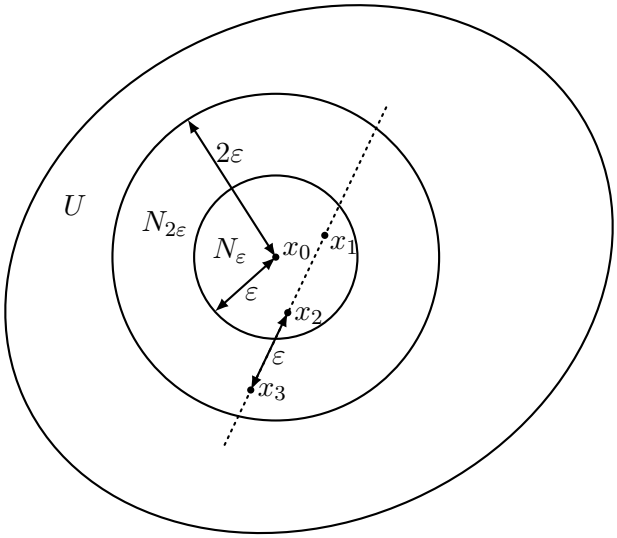


Figure 2.9: Construction considered in Theorem 2.28.

Theorem 2.28
Let f be convex and $U \subseteq \operatorname{dom} f$ an open set. If f is bounded from above in a neighbourhood of one point U, then f is locally Lipschitz in U.

Proof: By Theorem 2.26, f is locally bounded. Given x_0, we may find a neighbourhood $N_{2\varepsilon}(x_0) \subseteq U$ with radius 2ε on which f is bounded, say by M. Then f satisfies the stated Lipschitz condition on $N_\varepsilon(x_0)$, for if it does not, we may choose $x_1, x_2 \in N_\varepsilon(x_0)$ such that

$$\begin{aligned} f(x_2) - f(x_1) &> \frac{2M}{\varepsilon}\|x_2 - x_1\| \\ \Leftrightarrow \frac{f(x_2) - f(x_1)}{\|x_2 - x_1\|} &> \frac{2M}{\varepsilon} \end{aligned} \tag{2.55}$$

and $\alpha > 0$ such that $x_3 = x_2 + \alpha(x_2 - x_1)$ is still in $N_{2\varepsilon}(x_0)$ with $\|x_2 - x_3\| = \varepsilon$. Figure 2.9 is a visual depiction of this construction. By the convexity of f and Proposition 2.19, it follows that

$$\frac{f(x_3) - f(x_2)}{\|x_3 - x_2\|} \geqslant \frac{f(x_2) - f(x_1)}{\|x_2 - x_1\|} > \frac{2M}{\varepsilon} \tag{2.56}$$

This implies that $f(x_3) - f(x_2) > 2M$, which is in contradiction with $|f(x)| \leqslant M$ on $N_{2\varepsilon}(x_0)$. □

Theorem 2.29
Let f be convex and $U \subseteq \operatorname{dom} f$ an open set. If f is bounded from above in a neighbourhood of one point of U, then f is continuous on U.

PROOF: Theorem 2.28 implies that f is locally Lipschitz, from which continuity follows immediately. □

Theorem 2.30
Assume f is a convex function on $\mathbb{R}^n$. Then f is continuous on $\operatorname{int}(\operatorname{dom} f)$.

PROOF: If we consider the convex combination $x = \sum_{i=1}^{n} \lambda_i x_i$ with $x_i \in \operatorname{dom} f \ \forall i$, then it follows from the convexity of f that

$$f(x) \leqslant \sum_{i=1}^{n} \lambda_i f(x_i) \leqslant \left(\sum_{i=1}^{n} \lambda_i \right) \max_i f(x_i) = \max_i f(x_i) < \infty \tag{2.57}$$

Now let us consider an arbitrary $x \in \operatorname{int}(\operatorname{dom} f)$ and choose any n-dimensional simplex $S = \operatorname{conv}(\{x_0, x_1, \ldots, x_n\})$ with $x \in \operatorname{int}(S)$ and $S \subseteq \operatorname{dom} f$. Then it follows from what we have just seen that for any y in S we have

$$f(y) \leqslant \max_i f(x_i) < \infty \tag{2.58}$$

and Theorem 2.29 states that f must be continuous on $\operatorname{int}(\operatorname{dom} f)$. □

Remark 2.31
A convex function can have discontinuities on the boundary of its domain. Consider for example the function $f : \mathbb{R} \to \mathbb{R}$ defined by

$$f(x) = \begin{cases} x^2 & x \in (-1, 1) \\ 2 & x \in \{-1, 1\} \\ +\infty & \text{else} \end{cases}$$

Then f is obviously not continuous in 1 and -1. Furthermore, Theorem 2.30 only holds in finite dimensional spaces. In infinite dimensional spaces there exist linear functions (which are of course also convex) that are not continuous.

2.3 Subdifferential calculus

Our next goal will be study of the subdifferential calculus. As we will see, subdifferentials represent a generalisation of gradients for non-differentiable functions. Three questions are of great importance for us. What exactly are subdifferentials? When do they exist? What properties do they have? All these questions will be answered in detail in this section.

The theory presented in this section is based on results from [17, 52, 69]. The results presented here in this section differ from the references in so far that we adapted the statements such that they hold for general normed vector spaces. In [17] certain results are stated for locally convex spaces, whereas [52, 69] restrict themself to finite dimensional spaces.

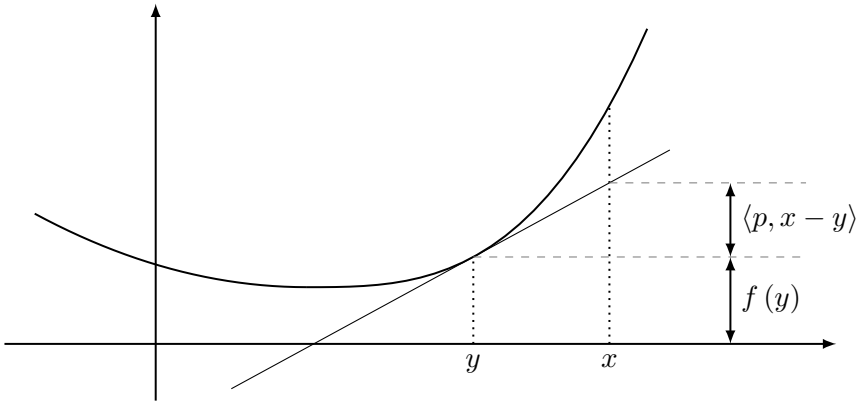


Figure 2.10: Geometric interpretation of the subgradient.

Definition 2.32 (Subdifferential and Subgradient)
The *subdifferential* of f at $\bar{x} \in X$ is defined as

$$\partial f(\bar{x}) := \{x^* \in X^* \mid f(x) - f(\bar{x}) \geqslant \langle x^*, x - \bar{x} \rangle \ \forall x \in X\}$$

The elements $x^* \in \partial f(\bar{x})$ are called *subgradients*. If $f(\bar{x}) = \pm\infty$, then we set the subdifferential of f at that position to $\partial f(\bar{x}) = \emptyset$.

Before continuing with the theory of subdifferentials, let us consider a few simple examples. It is a well known fact that a convex and differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ verifies for all $x, y \in \mathbb{R}^n$ the following equation

$$f(x) \geqslant f(y) + \langle \nabla f(y), x - y \rangle \tag{2.59}$$

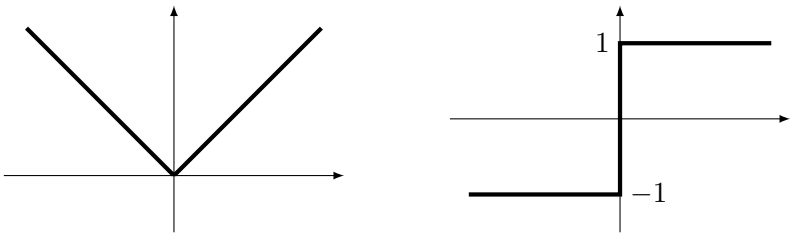
As a consequence the classical gradient is a subgradient for convex and differentiable functions.

Figure 2.10 shows us a nice geometric interpretation of the subgradient for functions from $\mathbb{R}$ to $\mathbb{R}$. The thin straight line with slope p touches the graph of f only at the point y and remains below the graph at every other point. As we see from the picture, we have

$$f(x) \geqslant f(y) + \langle p, x - y \rangle$$

for every point x. Definition 2.32 now implies that p is a subgradient of f at y. This means that in general, for functions from $\mathbb{R}$ to $\mathbb{R}$, the subgradients are the slopes of the lines that touch a function at one point at least and remain always below the graph of the function. Note that the subgradient need not to be unique. The function $f(x) := |x|$ has a non-unique subgradient in 0. It is easy to verify that its subdifferential is given by

$$\partial f(x) = \begin{cases} -1 & x < 0 \\ [-1, 1] & x = 0 \\ 1 & x > 0 \end{cases}$$



(a) Plot of $f(x) = |x|$. (b) Corresponding subgradients.

Figure 2.11: A function and its corresponding subgradients.

In this example we see that the subgradient is unique at all points where f is differentiable. In 0 however, f is not differentiable and we also lose the uniqueness of the subgradient. This is an interesting fact that is true in a more general setting as we are going to see later.

Furthermore, it should be clear, that subgradients do not necessarily always exist. Strictly concave functions from $\mathbb{R}$ to $\mathbb{R}$ obviously do not have subgradients. Also functions that are not continuous often lack subgradients at their points of discontinuity. Consider for example

$$f(x) := \begin{cases} 1 & x \geqslant 0 \\ 0 & x < 0 \end{cases} \tag{2.60}$$

Then $\partial f(0) = \emptyset$. Interestingly, if we change the definition of that function in 0 to the following function

$$f(x) := \begin{cases} 1 & x > 0 \\ 0 & x \leqslant 0 \end{cases} \tag{2.61}$$

then $\partial f(0) = \{0\}$. It follows that even if there is a discontinuity, there can still exist a subgradient.

Finally, we know that, for differentiable functions, the points at which the derivative vanishes, mark possible extrema of the function. Now if we look again at the subgradient of the absolute value, we see that 0 is a subgradient at $x = 0$. Interestingly this is also the point where we attain the minimum. Whether this is coincidence or not, will also be investigated in this section.

The first theorem that we will discuss now concerns the existence of subgradients for convex functions. This result as well as all the other statements up to Theorem 2.39 stem from [17]. The only significant difference between the results stated here and the ones found in [17] lies in the definition of the Gâteaux differential. We will use the definition found in [91] which requires it to be continuous. This assumption simplifies the argumentation at certain points. Furthermore, some proofs given here are presented with more details compared to [17], but the general approach remains identical.

Theorem 2.33
Assume f is a proper convex function. If f is continuous in $x \in X$ and $x \in \operatorname{int}(\operatorname{dom} f)$, then $\partial f(x) \neq \emptyset$.

PROOF: Since f is continuous in x, it follows from Proposition 2.16 that $\operatorname{int}(\operatorname{Epi} f)$ is non-empty. Furthermore, $\operatorname{Epi} f$ and $\operatorname{int}(\operatorname{Epi} f)$ are convex sets because of Proposition 2.8 and Proposition 2.12. It is also clear that $(x, f(x)) \notin \operatorname{int}(\operatorname{Epi} f)$, since $(x, f(x) - \delta) \notin \operatorname{Epi} f$ for all $\delta > 0$. From Proposition 2.3 it follows that we can separate the sets $\{(x, f(x))\}$ and $\operatorname{int}(\operatorname{Epi} f)$ through a linear functional. So there exists $z^* \in (X \times \mathbb{R})^*$, $z^* \not\equiv 0$ such that

$$\langle z^*, (x, f(x)) \rangle \leqslant \langle z^*, z \rangle \tag{2.62}$$

for all $z \in \operatorname{Epi} f$. We now decompose z^* in the following way

$$\langle z^*, (\xi, \alpha) \rangle = \langle x^*, \xi \rangle + \lambda\alpha \tag{2.63}$$

with $x^* \in X^*$ and $\lambda \in \mathbb{R}$. λ cannot be 0, otherwise we would have

$$\langle x^*, x \rangle \leqslant \langle x^*, \xi \rangle \quad \forall \xi \in \operatorname{dom} f \tag{2.64}$$

But since x is an inner point of $\operatorname{dom} f$, there exists for any $h \in X$ a $\delta > 0$ such that we have $x + \delta h \in \operatorname{dom} f$ and $x - \delta h \in \operatorname{dom} f$. It follows for $\xi = x \pm \delta h$,

$$\langle x^*, x \rangle \leqslant \langle x^*, x + \delta h \rangle \Leftrightarrow \langle x^*, h \rangle \geqslant 0 \quad \forall h \in X \tag{2.65}$$

$$\langle x^*, x \rangle \leqslant \langle x^*, x - \delta h \rangle \Leftrightarrow \langle x^*, h \rangle \leqslant 0 \quad \forall h \in X \tag{2.66}$$

This would imply $x^* \equiv 0$ which contradicts $z^* \not\equiv 0$. Therefore, we have $\lambda \neq 0$. Now it follows from eq. (2.62), that for all $\xi \in \operatorname{dom} f$ we get with $z = (\xi, f(\xi))$

$$\langle x^*, x \rangle + \lambda f(x) \leqslant \langle x^*, \xi \rangle + \lambda f(\xi) \tag{2.67}$$

Which is equivalent to

$$f(\xi) - f(x) \geqslant \langle -\frac{1}{\lambda} x^*, \xi - x \rangle \tag{2.68}$$

for all $\xi \in \operatorname{dom} f$. If $\xi \notin \operatorname{dom} f$, then the lefthand side must be $+\infty$ because we assumed f to be proper convex. Thus the inequality also remains verified in this case and therefore, we have

$$-\frac{1}{\lambda} x^* \in \partial f(x) \tag{2.69}$$

□

The previous theorem shows the importance of Theorem 2.30. It guarantees the existence of subgradients for convex functions with $\operatorname{dom} f = \mathbb{R}^n$ and it states that for the general case, where $f : X \to \mathbb{R}$ with an arbitrary space X, all we need is to require local boundedness.

Now that we have proven the existence of subgradients, we would like to know what additional constraints are necessary to have uniqueness. The uniqueness of the subgradient is closely linked to the concept of differentiability. Therefore, we start with introducing a certain number of concepts that generalise the classical notion of differentiability. Our goal will be the definition of the so called Gâteaux differential.

Lemma 2.34
Let $f : \mathbb{R} \to \mathbb{R} \cup \{+\infty\}$ be a convex function and $x \in \operatorname{dom} f$. Then

$$d(t) := \frac{f(x+t) - f(x)}{t} \tag{2.70}$$

is a monotonically increasing function on $(0, \infty)$ with values in $\overline{\mathbb{R}}$. Furthermore, we have $d(-t) \leqslant d(t)$ for all $t > 0$.

PROOF: For $0 < s < t$ we have

$$x + s = \frac{t-s}{t} x + \frac{s}{t}(x+t) \tag{2.71}$$

and because of the convexity of f it follows that

$$f(x+s) \leqslant \frac{t-s}{t} f(x) + \frac{s}{t} f(x+t) \tag{2.72}$$

Subtracting $f(x)$ and then dividing by $s > 0$, implies

$$d(s) = \frac{f(x+s) - f(x)}{s} \leqslant \frac{f(x+t) - f(x)}{t} = d(t) \tag{2.73}$$

which yields the monotonicity. Furthermore, for $t > 0$ we have

$$f(x) \leqslant \frac{1}{2} f(x-t) + \frac{1}{2} f(x+t) \tag{2.74}$$

and thus

$$f(x) - f(x-t) \leqslant f(x+t) - f(x) \tag{2.75}$$

If we divide by t, we obtain

$$\frac{f(x-t) - f(x)}{-t} \leqslant \frac{f(x+t) - f(x)}{t} \tag{2.76}$$

which is the desired inequality. □

Definition 2.35 (Directional derivative)
Let f be a function from X to $\overline{\mathbb{R}}$ and $x, h \in X$. If the limit

$$f'(x; h) := \lim_{t \searrow 0} \frac{f(x+th) - f(x)}{t} \tag{2.77}$$

exists, then we call it the *directional derivative* of f at x in direction h. It describes the rate at which a function f changes at the point x in direction h.

Proposition 2.36
The directional derivative is positive homogeneous, i.e. we have

$$f'(x; sh) = s f'(x; h) \quad x, h \in X, s > 0 \tag{2.78}$$

PROOF: By definition we have

$$\begin{aligned} f'(x;sh) &= \lim_{t\searrow 0} \frac{f(x+tsh)-f(x)}{t} \\ &= \lim_{t\searrow 0} \frac{f(x+tsh)-f(x)}{st} s \\ &= s \cdot \lim_{st\searrow 0} \frac{f(x+tsh)-f(x)}{st} \\ &= s f'(x;h) \end{aligned}$$

□

Theorem 2.37
Assume $f : X \to \overline{\mathbb{R}}$ is a proper convex function and $x \in \operatorname{dom} f$. Then the directional derivative exists for all $h \in X$ and we have

$$f'(x;h) = \inf_{t>0} \frac{f(x+th)-f(x)}{t} \tag{2.79}$$

Furthermore, we have

$$f'(x;h) \leqslant f(x+h) - f(x) \tag{2.80}$$

and

$$-f'(x;-h) \leqslant f'(x;h) \tag{2.81}$$

PROOF: We apply Lemma 2.34 on $\phi(t) := f(x+th)$. Then the difference quotient

$$d_h(t) = \frac{\phi(t)-\phi(0)}{t} = \frac{f(x+th)-f(x)}{t} \tag{2.82}$$

is monotonically increasing on $(0,\infty)$, and therefore, $\lim_{t\downarrow 0} d_h(t)$ exists and is equal to $\inf_{t>0} d_h(t)$. Equation (2.80) follows immediately from $d_h(1) = f(x+h) - f(x)$ and eq. (2.81) follows from Lemma 2.34 and

$$-\frac{f(x-th)-f(x)}{t} = d_h(-t) \leqslant d_h(t) = \frac{f(x+th)-f(x)}{t} \tag{2.83}$$

by considering the limit when $t \downarrow 0$. □

Definition 2.38 (Gâteaux differential)
Assume $M \subseteq X$ an open subset with $f : M \to \overline{\mathbb{R}}$ and $x \in M$. If the directional derivative $f'(x;h)$ exists for all $h \in X$ and if the mapping $h \mapsto f'(x;h)$ is linear and continuous, then we call the function $f'(x) : X \to \mathbb{R}$ defined by

$$\langle f'(x), h\rangle := f'(x;h) \tag{2.84}$$

the *Gâteaux differential* of f in x. If f has a Gâteaux differential in every point of M, then we say that f is *Gâteaux differentiable* on M.

The Gâteaux differentials are the generalisation of the partial derivatives for functions in $\mathbb{R}^n$. As the following theorem suggests, their existence is essential for the uniqueness of the subgradient.

Note that Theorem 2.37 does not imply that every proper convex function is Gâteaux differentiable. In order to be Gâteaux differentiable, the directional derivative $f'(x;h)$ has to be a linear and continuous map. These properties do not follow from Theorem 2.37.

Theorem 2.39
Let f be a proper convex function on X, $x \in \operatorname{int}(\operatorname{dom} f)$ and assume that f has a Gâteaux derivative $f'(x)$ in x. Then it follows that $\partial f(x) = \{f'(x)\}$.

PROOF: We apply Theorem 2.37. Equation (2.80) implies that we have

$$\langle f'(x), y - x\rangle = f'(x; y - x) \leqslant f(y) - f(x) \quad \forall y \in X \tag{2.85}$$

and therefore, we immediately have $f'(x) \in \partial f(x)$. Assume now that $x^* \in \partial f(x)$. The definition of the subgradient implies that

$$f(x + th) - f(x) \geqslant \langle x^*, th\rangle = t\langle x^*, h\rangle \tag{2.86}$$

for all $h \in X$ and $t > 0$. From this we conclude

$$\langle x^*, h\rangle \leqslant \frac{f(x + th) - f(x)}{t} \quad \forall h \in X, t > 0 \tag{2.87}$$

Considering the limit when $t \searrow 0$ we obtain

$$\langle x^*, h\rangle \leqslant \langle f'(x), h\rangle \Leftrightarrow \langle f'(x) - x^*, h\rangle \geqslant 0 \tag{2.88}$$

for all $h \in X$. By considering h and $-h$ we can conclude that $f'(x) = x^*$ □

Corollary 2.40
Assume $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function and $y \in \operatorname{int}(\operatorname{dom} f)$. If f is differentiable in y, then $\nabla f(y)$ is the only subgradient of f in y.

PROOF: This is an immediate consequence from Theorem 2.39. The classical notion of differentiability in $\mathbb{R}^n$ is a special case of the Gâteaux differential and satisfies all requirements from Theorem 2.39. □

So far, all our statements made the assumption that f was a convex function and we were then able to make claims about the existence and uniqueness of subgradients. We will now briefly examine if it is also possible to make statements in the other direction. Can we make statements about f if we know for example that the subgradient exists at every point? Before we answer this question, we will shortly present an interesting link between the definition of the subdifferential and the definition of strict convexity. We will then be able to give results about convexity and strict convexity simultaneously in the statements that will follow afterwards. Propositions 2.41 and 2.43 and Theorem 2.42, presented hereafter, have already been discussed in [52] in the context of convex optimisation. The formulations given here are almost identical. We will, however, present Proposition 2.43 in a slightly more general form by using the Gâteaux differentials which we introduced earlier.

Proposition 2.41
Assume f is strictly convex on X. Then, for every subgradient x^* at $x_0 \in \operatorname{dom} f$, we have

$$f(x) > f(x_0) + \langle x^*, x - x_0 \rangle \quad \forall x \in \operatorname{dom} f \tag{2.89}$$

Proof: By definition of the subgradient, we have

$$f(x) \geqslant f(x_0) + \langle x^*, x - x_0 \rangle \quad \forall x \in X \tag{2.90}$$

Assume now that there exists a $z \in \operatorname{dom} f$ such that we have equality between the two expressions.

$$f(z) = f(x_0) + \langle x^*, z - x_0 \rangle \tag{2.91}$$

Because of the strict convexity we have for every $\lambda \in (0,1)$

$$\begin{aligned} f(\lambda x_0 + (1-\lambda) z) &< \lambda f(x_0) + (1-\lambda) f(z) \\ &= f(x_0) + (1-\lambda) \langle x^*, z - x_0 \rangle \end{aligned} \tag{2.92}$$

The definition of the subgradient now implies for $x = \lambda x_0 + (1-\lambda) z$ that we have

$$\begin{aligned} f(\lambda x_0 + (1-\lambda) z) &\geqslant f(x_0) + \langle x^*, \lambda x_0 + (1-\lambda) z - x_0 \rangle \\ &\geqslant f(x_0) + (1-\lambda) \langle x^*, z - x_0 \rangle \end{aligned} \tag{2.93}$$

This contradicts the previous equation. Thus such a z cannot exist. Therefore,

$$f(x) > f(x_0) + \langle x^*, x - x_0 \rangle \quad \forall x \in \operatorname{dom} f \tag{2.94}$$

□

Theorem 2.42
Assume $f : X \to \mathbb{R}$ is a function and $S \subseteq X$ a convex set. If there exists a subgradient of f at every point $y \in \operatorname{int}(S)$, then f is convex on $\operatorname{int}(S)$. Furthermore, if the strict inequality holds, then f is even strictly convex on $\operatorname{int}(S)$.

Proof: Assume y_1 and y_2 are two points inside $\operatorname{int}(S)$. Proposition 2.8 implies that $\operatorname{int}(S)$ must also be convex and therefore, $\lambda y_1 + (1-\lambda) y_2 \in \operatorname{int}(S)$ for all $\lambda \in (0,1)$. Assume now that x^* is a subgradient in such a point (for any λ). The we have

$$f(y_1) \geqslant f(\lambda y_1 + (1-\lambda) y_2) + (1-\lambda) \langle x^*, y_1 - y_2 \rangle \tag{2.95}$$

$$f(y_2) \geqslant f(\lambda y_1 + (1-\lambda) y_2) + \lambda \langle x^*, y_2 - y_1 \rangle \tag{2.96}$$

If we multiply these two inequalities with λ (resp. $1-\lambda$) and add them, we receive

$$\lambda f(y_1) + (1-\lambda) f(y_2) \geqslant f(\lambda y_1 + (1-\lambda) y_2) \tag{2.97}$$

and therefore, f is convex. The same argumentation proves the statement about the strict convexity if we exchange "$\geqslant$" with "$>$". □

Proposition 2.43
Let $f : X \to \mathbb{R}$ be a function, $S \subseteq X$ open and convex and assume that f has a Gâteaux derivative $f'(x)$ in every x in S. Then f is convex on S if and only if for all x, y in S the following inequality holds

$$f(x) \geqslant f(y) + \langle f'(y), x - y \rangle \tag{2.98}$$

If we have

$$f(x) > f(y) + \langle f'(y), x - y \rangle \tag{2.99}$$

for all $x \neq y$ in S, then f is even strictly convex.

Proof: From Theorem 2.39 it follows that if f is convex and has a Gâteaux derivative in X^* at every point y in S, then $f'(y)$ is the unique subgradient. Therefore, eq. (2.98) must hold. The strict inequality is a direct consequence of Proposition 2.41. On the other side if eq. (2.98) holds, then $f'(y)$ is a subgradient of f in every $y \in S$ and it follows from Theorem 2.42 that f must be convex on S, respectively strictly convex if the strict inequality holds. □

If we are in $\mathbb{R}^n$, then the previous proposition simply states that a differentiable function is convex if and only if

$$f(x) \geqslant f(y) + \langle \nabla f(y), x - y \rangle$$

holds for all $x, y \in \mathbb{R}^n$. Thus, we have proven the well known formula that we stated in the beginning of this section.

We now know that the subgradient is a generalisation of the gradient and we are able to state under which conditions it exists and when we can expect uniqueness. Our next goal will be to study how far the other properties of the gradient extend to subgradients. The most important question is certainly whether we can still make statements about the existence of extrema of a function similarly as with the help of gradients. Also we know that gradients are linear, i.e. they fulfill

$$\nabla (f + g)(x_0) = \nabla f(x_0) + \nabla g(x_0) \tag{2.100}$$

The next few propositions and theorems will show, that under certain conditions, similar results also hold for subgradients. If our functions behave sufficiently well, then we have

$$\partial (f + g)(x_0) = \partial f(x_0) + \partial g(x_0) \tag{2.101}$$

This highly useful property will be of great importance later on and therefore, it is necessary to study the exact conditions under which this equality holds. This will be done in Proposition 2.44 and Theorem 2.45. Another similar result related to the composition with linear maps will be discussed in Theorem 2.46. Our proofs of these three statements closely follow the argumentation presented in [17]. However, a few steps were reformulated such that it becomes possible to reuse some of the results from this thesis. Furthermore, we added a few details to facilitate the lecture of the proofs.

Proposition 2.44
Assume that $y \in \operatorname{dom} f \cap \operatorname{dom} g$, $p \in \partial f(y)$ and $q \in \partial g(y)$. Then $p + \lambda q \in \partial (f + \lambda g)(y)$ for all $\lambda \geqslant 0$.

Proof: By definition we have:

$$f(x) - f(y) \geqslant \langle p, x - y \rangle \quad \forall x \in X$$
$$g(x) - g(y) \geqslant \langle q, x - y \rangle \quad \forall x \in X$$

Adding the two equations gives:

$$(f + \lambda g)(x) - (f + \lambda g)(y) \geqslant \langle p + \lambda q, x - y \rangle \quad \forall x \in X,\ \lambda \geqslant 0$$

And therefore, $p + \lambda q \in \partial (f + \lambda g)(y)$ □

Theorem 2.45 (Subdifferential sum formula)
Assume f_1, f_2 are two proper convex functions on X. Furthermore, let there be a vector $\bar{x} \in \operatorname{int}(\operatorname{dom} f_1) \cap \operatorname{dom} f_2$ and assume f_1 is continuous in $\bar{x}$. Then we have

$$\partial (f_1 + f_2)(x) = \partial f_1(x) + \partial f_2(x) \tag{2.102}$$

for all $x \in \operatorname{dom} f_1 \cap \operatorname{dom} f_2$.

Proof: We already know from Proposition 2.44 that the inclusion

$$\partial (f_1 + f_2)(x) \supseteq \partial f_1(x) + \partial f_2(x) \quad \forall x \in \operatorname{dom} f_1 \cap \operatorname{dom} f_2 \tag{2.103}$$

is true. Therefore, it is enough to show the inverse inclusion. Choose $x \in \operatorname{dom} f_1 \cap \operatorname{dom} f_2$ and assume $x^* \in \partial (f_1 + f_2)(x)$. We have to show that there exist x_1^* and x_2^* such that $x^* = x_1^* + x_2^*$ with $x_1^* \in \partial f_1(x)$ and $x_2^* \in \partial f_2(x)$. By the definition of $\partial (f_1 + f_2)(x)$, we have

$$f_1(y) - f_1(x) - \langle x^*, y - x \rangle \geqslant f_2(x) - f_2(y) \quad \forall y \in \operatorname{dom} f_1 \cap \operatorname{dom} f_2 \tag{2.104}$$

Consider now the following sets

$$C_1 := \{(y, \alpha) \mid f_1(y) - f_1(x) - \langle x^*, y - x \rangle \leqslant \alpha\} \tag{2.105}$$
$$C_2 := \{(y, \alpha) \mid \alpha \leqslant f_2(x) - f_2(y)\} \tag{2.106}$$

Since f_1 and f_2 are convex functions, it is easy to see that these sets are again convex. But C_1 is also the epigraph of the convex function

$$g(y) := f_1(y) - f_1(x) - \langle x^*, y - x \rangle \tag{2.107}$$

From Proposition 2.16 it follows that the interior of C_1 is non-empty, since $\bar{x} \in \operatorname{int}(\operatorname{dom} g)$ and g is continuous in $\bar{x}$. We will now show that $\operatorname{int}(C_1) \cap C_2 = \emptyset$. Take any (y, β) from $\operatorname{int}(C_1)$, then there exists $\varepsilon > 0$ such that $(y, \beta - \varepsilon) \in C_1$. Equation (2.104) implies that

$$g(z) \geqslant f_2(x) - f_2(z) \quad \forall z \in \operatorname{dom} f_1 \cap \operatorname{dom} f_2 \tag{2.108}$$

From this equation we conclude that for our $(y, \beta) \in \operatorname{int}(C_1)$ we have

$$\begin{aligned} & f_2(y) \geqslant f_2(x) - g(y) \\ \Leftrightarrow\ & f_2(y) + \beta \geqslant f_2(x) - g(y) + \beta \\ & \geqslant f_2(x) - (\beta - \varepsilon) + \beta \quad (\text{because } g(y) \leqslant (\beta - \varepsilon)) \\ & = f_2(x) + \varepsilon \end{aligned} \tag{2.109}$$

This implies that $f_2(y)+\beta \geqslant f_2(x)+\varepsilon$ and thus $f_2(x)-f_2(y) < \beta$. Therefore, $(y, \beta) \notin C_2$. It follows that we have two sets $\operatorname{int}(C_1)$ and C_2, where $\operatorname{int}(C_1)$ is by definition open, which are disjoint. Proposition 2.3 implies there exists a $z^* \in (X \times \mathbb{R})^*$ and a $\alpha \in \mathbb{R}$, such that

$$\langle z^*, c_1 \rangle < \alpha \leqslant \langle z^*, c_2 \rangle \quad \forall\ c_1 \in \operatorname{int}(C_1),\ c_2 \in C_2 \tag{2.110}$$

and $\langle z^*, c_1 \rangle \leqslant \alpha$ for all $c_1 \in C_1$. This z^* can be rewritten in the in the following form

$$\langle z^*, (x, \beta) \rangle = \langle y^*, x \rangle + \lambda\beta \tag{2.111}$$

where $y^* \in X^*$ and $\lambda \in \mathbb{R}$. We notice, that $(x, 0) \in C_1 \cap C_2$. Plugging this point in eq. (2.110) implies that $\alpha = \langle y^*, x \rangle$ and therefore, we have

$$\langle y^*, y \rangle + \lambda\beta \leqslant \langle y^*, x \rangle \leqslant \langle y^*, z \rangle + \lambda\gamma \tag{2.112}$$

for all $(y, \beta) \in C_1$ and $(z, \gamma) \in C_2$. We will now show that $\lambda < 0$ must hold. Clearly $(\bar{x}, g(\bar{x}) + 1) \in \operatorname{int} C_1$ and $(\bar{x}, f_2(x) - f_2(\bar{x})) \in C_2$. Again by eq. (2.110), we obtain

$$\langle y^*, \bar{x} \rangle + \lambda(g(\bar{x}) + 1) < \langle y^*, \bar{x} \rangle + \lambda(f_2(x) - f_2(\bar{x})) \tag{2.113}$$

So that we have

$$\lambda \underbrace{(g(\bar{x}) - f_2(x) + f_2(\bar{x}) + 1)}_{>0 \text{ by eq. (2.108)}} < 0 \tag{2.114}$$

Equation (2.108) now implies that $\lambda < 0$ must hold. This allows us now to consider the following decomposition

$$x^* = \left(x^* - \frac{1}{\lambda} y^*\right) + \frac{1}{\lambda} y^* \tag{2.115}$$

We will show that

$$x^* - \frac{1}{\lambda} y^* \in \partial f_1(x) \text{ and } \frac{1}{\lambda} y^* \in \partial f_2(x)$$

It is clear that $(y, g(y)) \in C_1$ and $(z, f_2(x) - f_2(z)) \in C_2$ for all $y \in \operatorname{dom} f_1 = \operatorname{dom} g$ and $z \in \operatorname{dom} f_2$. Using the first half eq. (2.112) it follows that

$$\begin{aligned} & \langle y^*, y \rangle + \lambda g(y) \leqslant \langle y^*, x \rangle \\ \Leftrightarrow\ & \langle y^*, y \rangle + \lambda(f_1(y) - f_1(x) - \langle x^*, y - x \rangle) \leqslant \langle y^*, x \rangle \\ & \Leftrightarrow f_1(x) - f_1(y) \geqslant \langle x^* - \tfrac{1}{\lambda} y^*, x - y \rangle \end{aligned} \tag{2.116}$$

As a consequence $x^* - \frac{1}{\lambda} y^* \in \partial f_1(x)$. And similarly the second half of eq. (2.112) implies that

$$\begin{aligned} & \langle y^*, x \rangle \leqslant \langle y^*, y \rangle + \lambda \left(f_2(x) - f_2(y) \right) \\ \Leftrightarrow\ & f_2(x) - f_2(y) \geqslant \langle \tfrac{1}{\lambda} y^*, x - y \rangle \end{aligned} \tag{2.117}$$

and therefore, $\frac{1}{\lambda} y^* \in \partial f_2(x)$. This concludes the proof. □

It is clear that the previous theorem also applies to sums with more than two functions. We just have to apply it recursively. In [37] the authors showed that the previous theorem remains true under slightly weaker requirements. It is enough if the space on which our functions operate is locally convex. It does not need to be a normed vector space. However, the formulation that we considered here is simpler to prove and will be sufficient for our needs. The more general proof from [37] requires a generalisation of the Theorem of Hahn-Banach (Theorem 2.1) for locally convex spaces. Such a generalisation can be found in [37, 91].

The next theorem will state a similar result as the subdifferential sum formula, but this time, for the composition with linear maps.

Theorem 2.46 (Subdifferential for composition with linear maps)
Assume X, Y are normed vector spaces over $\mathbb{R}$, $A : X \to Y$ a continuous linear mapping and f a proper convex function on Y. Furthermore, assume that the range $\mathcal{R}(A)$ of A is a subset of $\operatorname{dom} f$ and that there exists $\bar{x} \in X$ such that $A\bar{x} \in \operatorname{int}(\operatorname{dom} f)$ and that f is continuous in $A\bar{x}$. Then we have for all $x \in X$

$$\partial (f \circ A)(x) = A^* \partial f(Ax) \tag{2.118}$$

where A^* is the adjoint operator of A.

Proof: "$\supseteq$": Assume $y^* \in \partial f(Ax)$. Then we have by definition of the subdifferential

$$f(y) \geqslant f(Ax) + \langle y^*, y - Ax \rangle \quad \forall y \in Y \tag{2.119}$$

and therefore, also

$$f(A\xi) \geqslant f(Ax) + \langle y^*, A\xi - Ax \rangle \quad \forall \xi \in X \tag{2.120}$$

This implies that

$$f(A\xi) \geqslant f(Ax) + \langle A^* y^*, \xi - x \rangle \quad \forall \xi \in X \tag{2.121}$$

Thus we have $A^* y^* \in \partial (f \circ A)(x)$.

"$\subseteq$": Assume now that $x \in \operatorname{dom}(f \circ A)$ and $x^* \in \partial (f \circ A)(x)$. Define the subset $U \subseteq Y \times \mathbb{R}$ by

$$U := \{ (A\xi, f(Ax) + \langle x^*, \xi - x \rangle) \mid \xi \in X \} \tag{2.122}$$

Since $A\bar{x}$ is an inner point of $\operatorname{dom} f$ and f is continuous in $A\bar{x}$ it follows from Proposition 2.16 that $\operatorname{int}(\operatorname{Epi} f)$ is non-empty. Furthermore, we have $U \cap \operatorname{int}(\operatorname{Epi} f) = \emptyset$, because if we assume that $(y, \beta) \in \operatorname{int}(\operatorname{Epi} f)$, then there exists $\varepsilon > 0$ sufficiently small, such that

$(y, \beta - \varepsilon)$ is still an element of $\operatorname{Epi} f$. If (y, β) were an element of U, then there would have to exist a $\xi \in X$ such that $y = A\xi$ and $\beta = f(Ax) + \langle x^*, \xi - x \rangle$. Since (y, β) lies in the epigraph, this ξ fulfills the inequality

$$f(A\xi) \leqslant f(Ax) + \langle x^*, \xi - x \rangle - \varepsilon \tag{2.123}$$

which is a contradiction to $x^* \in \partial(f \circ A)(x)$. Thus, the sets must be disjoint. Proposition 2.3 now implies that there exists $z^* \in (Y \times \mathbb{R})^*$ and an $\alpha \in \mathbb{R}$. such that

$$\langle z^*, u \rangle \leqslant \alpha \leqslant \langle z^*, e \rangle \tag{2.124}$$

for all $u \in U$ and $e \in \operatorname{Epi} f$ and $\alpha < \langle z^*, e \rangle$ for all inner points of $\operatorname{Epi} f$. We now rewrite

$$\langle z^*, (y, \beta) \rangle = \langle y^*, y \rangle + \lambda\beta \tag{2.125}$$

with $y^* \in Y^*$ and $\beta \in \mathbb{R}$. This implies that

$$\langle y^*, A\xi \rangle + \lambda f(Ax) + \lambda \langle x^*, \xi - x \rangle \leqslant \alpha \leqslant \langle y^*, y \rangle + \lambda\mu \tag{2.126}$$

must hold for all $\xi \in X$ and $(y, \mu) \in \operatorname{Epi} f$. Obviously λ cannot be 0, because if it were, we simply consider $\xi = \bar{x}$ and $(A\bar{x}, f(A\bar{x}) + 1) \in \operatorname{int}(\operatorname{Epi} f)$ and obtain from eq. (2.126)

$$\langle y^*, A\bar{x} \rangle \leqslant \alpha < \langle y^*, A\bar{x} \rangle \tag{2.127}$$

which is impossible. Furthermore, if we consider $y = A\xi$ and $\mu = f(A\xi) + 1$, then eq. (2.126) gives us

$$\begin{aligned} &\langle y^*, A\xi \rangle + \lambda f(Ax) + \lambda \langle x^*, \xi - x \rangle \leqslant \langle y^*, A\xi \rangle + \lambda (f(A\xi) + 1) \\ \Leftrightarrow\ &\lambda \underbrace{(f(A\xi) - f(Ax) - \langle x^*, \xi - x \rangle + 1)}_{>0} \geqslant 0 \end{aligned} \tag{2.128}$$

which implies that $\lambda > 0$ because the expression inside the brackets is positive since x^* is a subgradient. Finally the first half of eq. (2.126) can be rewritten as

$$\langle y^*, A\xi \rangle + \lambda \langle x^*, \xi \rangle + \lambda f(Ax) - \lambda \langle x^*, x \rangle \leqslant \alpha \tag{2.129}$$

Now assume that $\langle y^*, A\xi \rangle + \lambda \langle x^*, \xi \rangle \neq 0$. Then we could make it arbitrarily large by multiplying it with a constant. Thus we could find a ξ such that the inequality in eq. (2.129) does not hold anymore. Therefore, we must have

$$\langle y^*, A\xi \rangle + \lambda \langle x^*, \xi \rangle = 0 \tag{2.130}$$

From which we can conclude that

$$x^* = A^* \left(-\frac{1}{\lambda} y^* \right) \tag{2.131}$$

Applying eq. (2.130) and eq. (2.131) to eq. (2.126) leads to ($\mu = f(y)$)

$$\lambda f(Ax) - \lambda \langle x^*, x \rangle \leqslant \langle y^*, y \rangle + \lambda f(y) \tag{2.132}$$

for all $y \in \operatorname{dom} f$ so that we get

$$f(y) \geqslant f(Ax) + \langle -\frac{1}{\lambda} y^*, y - Ax \rangle \tag{2.133}$$

from which we can conclude that $-\frac{1}{\lambda} y^* \in \partial f(Ax)$. Because of eq. (2.131), it follows that

$$x^* \in A^* \partial f(Ax) \tag{2.134}$$

This concludes the proof. □

Similarly as for Theorem 2.45, it is also possible to generalise the previous theorem to locally convex spaces. The difficulties that arise are the same as for the subdifferential sum formula and the proof can again be found in [37].

Theorems 2.45 and 2.46 show us that subgradients share many characteristics with classical gradients. However, one of the most useful properties of gradients is that they can be used to find extrema of convex functions. So far we do not know whether this is possible with subdifferentials. The following proposition discusses this problem. It shows, that there exists a simple way to determine the minima of convex functions by looking at the subgradients. This is a highly useful observation and will greatly simplify the proofs given in Section 2.4. The formulation of Proposition 2.47, as it is given here, is a generalisation of a similar result found in [52], where it was discussed in the context of convex optimisation problems in $\mathbb{R}^n$. The proofs for both formulations are identical.

Proposition 2.47
For a convex function f, the following statements are equivalent.

1. There exists a subgradient x^* of f in x_0 that fulfills

$$\langle x^*, x - x_0 \rangle \geqslant 0 \quad \forall x \in X \tag{2.135}$$

2. $x_0 \in \operatorname{dom} f$ is a global minimum of f.

3. 0 is a subgradient of f in x_0.

Proof: (1) ⇒ (2): For a subgradient we have by definition

$$f(x) \geqslant f(x_0) + \langle x^*, x - x_0 \rangle \tag{2.136}$$

for all x in X. It immediately follows that when eq. (2.135) is fulfilled, then x_0 must be a global minimum.

(2) ⇒ (3): If x_0 is a global minimum, then $f(x) \geqslant f(x_0)$ for all x and $x^* \equiv 0$ is a valid subgradient.

(3) ⇒ (1): Assume $x^* \equiv 0$ is a subgradient. Then eq. (2.135) is trivially fulfilled. □

In later chapters (e.g. Section 2.4), we will frequently encounter the subgradient of a norm. Therefore, it would be handy to have an analytical expression for these subdifferentials. This problem has already been treated in [44]. The following proposition summarises their findings and presents a slightly more detailed proof as the one given in [44].

Proposition 2.48
$\|\cdot\|$ is subdifferentiable at every point in X and the subdifferential is given by

$$\partial\left(\|\cdot\|\right)(x) = \begin{cases} \{x^* \in X^* \mid \langle x^*,\ x\rangle = \|x\| \text{ and } \|x^*\|_\infty = 1\}, & \text{if } x \neq 0 \\ \{x^* \in X^* \mid \|x^*\|_\infty \leqslant 1\}, & \text{if } x = 0 \end{cases} \tag{2.137}$$

Proof: From Corollary 2.6, it follows immediately, that for all $x_0 \in X$, $x_0 \neq 0$, there exists a continuous linear functional $x_0^* \in X^*$ such that

$$\langle x_0^*,\ x_0\rangle = \|x_0\| \quad \text{and} \quad \|x_0^*\|_\infty = 1 \tag{2.138}$$

This implies that we have

$$\begin{aligned} \langle x_0^*,\ x - x_0\rangle &= \langle x_0^*,\ x\rangle - \langle x_0^*,\ x_0\rangle \\ &= \langle x_0^*,\ x\rangle - \|x_0\| \\ &\leqslant \|\langle x_0^*,\ x\rangle\| - \|x_0\| \\ &\leqslant \underbrace{\|x_0^*\|_\infty}_{=1}\|x\| - \|x_0\| \\ &= \|x\| - \|x_0\|, \ \forall x \in X \end{aligned} \tag{2.139}$$

which is equivalent to $x_0^* \in \partial\left(\|\cdot\|\right)(x_0)$. Conversely if we assume that $x_0^* \in \partial\left(\|\cdot\|\right)(x_0)$ holds, then from the definition of the subdifferential we can conclude that for all $x \in X$ we have

$$\|x_0\| - \|x\| \leqslant \langle x_0^*,\ x_0 - x\rangle \tag{2.140}$$

If we plug in $x = \lambda x_0$ with $\lambda \in \mathbb{R}$ and $\lambda > 0$ we obtain

$$(1-\lambda)\left(\langle x_0^*,\ x_0\rangle - \|x_0\|\right) \geqslant 0 \tag{2.141}$$

If $\lambda > 1$, then we must have $x_0\left(x_0\right) - \|x_0\| \leqslant 0$. Analogously $\lambda < 1$ would require $x_0\left(x_0\right) - \|x_0\| \geqslant 0$. Therefore, we must have

$$\langle x_0^*,\ x_0\rangle = \|x_0\| \tag{2.142}$$

and from eq. (2.140) it follows that $\langle x_0^*,\ x\rangle \leqslant \|x\|, \ \forall x \in X$ and thus $\|x_0^*\|_\infty = 1$. This yields the desired equality for $x_0 \neq 0$. If $x_0 = 0$ then it follows immediately from the definition of the subdifferential that we must have

$$\partial\left(\|\cdot\|\right)(x_0) = \{x^* \in X^* \mid x^*\left(x\right) \leqslant \|x\| \ \ \forall x \in X\} \tag{2.143}$$

which is equivalent to

$$\partial\left(\|\cdot\|\right)(x_0) = \{x^* \in X^* \mid \|x^*\|_\infty \leqslant 1\} \tag{2.144}$$

□

Corollary 2.49
The subdifferential of $\|\cdot\|_2 : \mathbb{R}^n \to \mathbb{R}$ is given by

$$\partial\left(\|\cdot\|_2\right)(x) = \begin{cases} \frac{x}{\|x\|_2}, & \text{if } x \neq 0 \\ \left\{y \in \mathbb{R}^n \mid \|y\|_2 \leqslant 1\right\}, & \text{if } x = 0 \end{cases} \tag{2.145}$$

Note that therefore, every subgradient $q \in \partial\left(\|\cdot\|_2\right)(x)$ fulfills $\|q\|_2 \leqslant 1$.

Proof: The statement follows immediately from Proposition 2.48 if one remembers that for $x \neq 0$, $\|\cdot\|_2$ is differentiable and its gradient is given by $\nabla \|x\|_2 = \frac{x}{\|x\|_2}$, which in this case coincides with the subdifferential. The case $x = 0$ follows directly from $(\mathbb{R}^n)^* \cong \mathbb{R}^n$ □

2.4 Shrinkage operators

In this section we will analyse two so called shrinkage operators. They are interesting for us for a very specific reason. The split Bregman algorithm that we will see later needs the minimiser of

$$\|x\|_1 + \frac{\lambda}{2}\|x - b\|_2^2 \tag{2.146}$$

an sometimes also the minimiser of

$$\|x\|_2 + \frac{\lambda}{2}\|x - b\|_2^2 \tag{2.147}$$

for some vector $b \in \mathbb{R}^n$ and $\lambda > 0$. Both problems have an analytical solution that can be expressed through shrinkage operators. The first one with the help of the so called soft shrinkage operator and the second one by using the generalised shrinkage operator.

Shrinkage operators are generally used in combination with wavelet decompositions for denoising purposes. See for example [59, 79], where the authors also presented other shrinkage operators such as the garrote and hard shrinkage operator. The proofs presented in this section will make heavy use of subgradients. They clearly illustrate the elegance and power of this concept of subdifferentiability.

2.4.1 Soft shrinkage

The first operator that we consider is the so called soft shrinkage operator. It is defined as follows:

Definition 2.50 (Soft shrinkage operator)
Assume $y \in \mathbb{R}$ and $\alpha > 0$. Then the *soft shrinkage operator* is given by

$$\operatorname{shrink}(y, \alpha) := \operatorname{sgn}(y) \cdot \max(|y| - \alpha, 0) = \begin{cases} y - \alpha & \text{if } y \in (\alpha, \infty) \\ 0 & \text{if } y \in [-\alpha, \alpha] \\ y + \alpha & \text{if } y \in (-\infty, -\alpha) \end{cases} \tag{2.148}$$

As already mentioned, the soft shrinkage operator is able to solve one of the above mentioned optimisation problems. First we will show that this is true in $\mathbb{R}$ and then we show that it easily extends to $\mathbb{R}^n$.

Proposition 2.51
Consider the function $f : \mathbb{R} \to \mathbb{R}$ defined by

$$f(x) = |x| + \frac{\lambda}{2}(x-b)^2 \tag{2.149}$$

where $\lambda > 0$ and $b \in \mathbb{R}$. Then f becomes minimal for $x = \text{shrink}\left(b, \frac{1}{\lambda}\right)$.

Proof: We note that f is a strictly convex function and thus every local minimum must necessarily also be a global minimum. Because of Proposition 2.47 it is enough to find those x for which 0 is an element of the subgradient of f. Theorem 2.45 implies that the subgradient of f is given by

$$\partial f(x) = \partial(|x|) + \lambda(x-b) \tag{2.150}$$

Therefore, x is a global minimiser if the following equation is fulfilled for some element $q \in \partial(|x|)$

$$0 = q + \lambda x - \lambda b \;\Leftrightarrow\; q = \lambda b - \lambda x \tag{2.151}$$

As we already know, there are only three possible choices for q. If $x > 0$ then $q = 1$, if $x < 0$ then $q = -1$ and if $x = 0$, then $q \in [-1, 1]$. Now if $\lambda b > 1$ then x must be positive because q cannot be larger than 1. But for positive x, $\partial(|x|) = q = 1$ and x must be $b - \frac{1}{\lambda}$. On the other side, if $\lambda b < -1$, then x must be negative, since q cannot be smaller than -1. For negative x, $\partial(|x|) = q = -1$ and x is given by $b + \frac{1}{\lambda}$. Now assume $\lambda b \in [0, 1]$, then $-\lambda x = q - \lambda b$ must hold. If we suppose $q > \lambda b$, then x would have to be strictly negative. However, for strictly negative x, $q = -1$ and we have a contradiction. In the same way we cannot have $q < \lambda b$ since then we must have $x > 0$ and $q = 1$. It follows that $q = \lambda b$ is the only remaining choice. This implies $x = 0$. The case $\lambda b \in [-1, 0]$ leads by identical reasoning to the same result $x = 0$ and $q = \lambda b$. This tells us that the minimiser is given by

$$x = \begin{cases} b - \frac{1}{\lambda} & \text{if } b \in \left(\frac{1}{\lambda}, \infty\right) \\ 0 & \text{if } b \in \left[-\frac{1}{\lambda}, \frac{1}{\lambda}\right] \\ b + \frac{1}{\lambda} & \text{if } b \in \left(-\infty, -\frac{1}{\lambda}\right) \end{cases} \tag{2.152}$$

which is exactly the definition of $\text{shrink}\left(b, \frac{1}{\lambda}\right)$. □

Corollary 2.52
Consider the function $f : \mathbb{R}^n \to \mathbb{R}$ defined by

$$f(x) = \|x\|_1 + \frac{\lambda}{2}\|x-b\|_2^2 \tag{2.153}$$

where $\lambda > 0$ and $b \in \mathbb{R}^n$. The minimiser of f is the vector whose components are given by $x_i = \text{shrink}\left(b_i, \frac{1}{\lambda}\right)$.

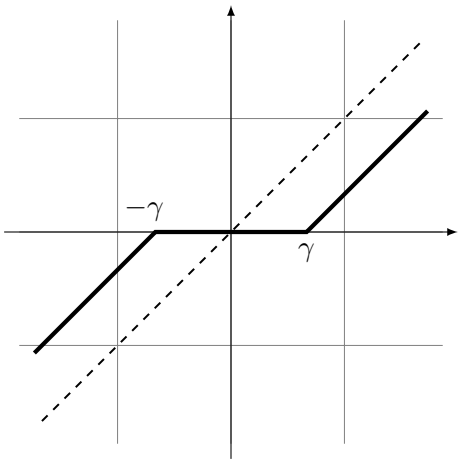


Figure 2.12: Graphical representation of $f(x) = x$ (dashed line) and $f(x) = \text{shrink}(x, \gamma)$ (thick line).

PROOF: $\|x\|_1 + \frac{\lambda}{2}\|x - b\|_2^2$ can be rewritten as

$$\|x\|_1 + \frac{\lambda}{2}\|x - b\|_2^2 = \sum_{i=1}^{n} |x_i| + \frac{\lambda}{2}(x_i - b_i)^2 \tag{2.154}$$

Obviously this sum is minimal, exactly when every term $|x_i| + \frac{\lambda}{2}(x_i - b_i)^2$ is minimal. Since Proposition 2.51 states that the minimiser of such an expression is given by $x_i = \text{shrink}\left(b_i, \frac{1}{\lambda}\right)$ the conclusion follows immediately. □

2.4.2 Generalised shrinkage

The second operator that we will consider is a generalisation of the soft shrinkage operator from Definition 2.50. We will simply call it generalised shrinkage operator and it is given by the following definition.

Definition 2.53 (Generalised shrinkage)
Let b be a vector in $\mathbb{R}^n$ and $\lambda > 0$, then we define the *generalised shrinkage operator* as

$$\text{gshrink}(b, \lambda) := \max(\|b\|_2 - \lambda, 0)\frac{b}{\|b\|_2} = \begin{cases} b - \frac{\lambda}{\|b\|_2} b, & \text{if } \|b\|_2 > \lambda \\ 0, & \text{else} \end{cases} \tag{2.155}$$

where we adopt the convention $0 \cdot \frac{0}{0} = 0$.

As in the case of the soft shrinkage operator, the generalised shrinkage operator allows us to solve a specific minimisation problem.

Proposition 2.54
Consider the function $f : \mathbb{R}^n \to \mathbb{R}$ given by

$$f(x) = \|x\|_2 + \frac{\lambda}{2}\|x - b\|_2^2 \tag{2.156}$$

with $b \in \mathbb{R}^n$ and $\lambda > 0$. Then f becomes minimal for $x^* = \text{gshrink}\left(b, \frac{1}{\lambda}\right)$.

PROOF: We will proceed by a similar reasoning as in Proposition 2.51. Since f is strictly convex, it has at most one minimiser. Thus x^* is a minimiser if and only if 0 is a subgradient of f at x^*. Since $\|\cdot\|_2^2$ is differentiable, it is enough to find an x^* that solves the following equation

$$0 = \lambda (x^* - b) + q \;\Leftrightarrow\; q + \lambda x^* = \lambda b \tag{2.157}$$

where q is an element of $\partial (\|x^*\|_2)$. Now if $\lambda \|b\|_2 > 1$, then x^* cannot be 0, since we know from Corollary 2.49 that the norm of q is at most 1. But for $x^* \neq 0$, we know that $q = \frac{x^*}{\|x^*\|_2}$. From this it follows that we have to solve

$$\lambda x^* \left(1 + \frac{1}{\lambda \|x^*\|_2}\right) = \lambda b \tag{2.158}$$

Taking the norm on both sides implies that $\|x^*\|_2 = \|b\|_2 - \frac{1}{\lambda}$. By plugging this expression back into our equation, we obtain

$$x^* \left(1 + \frac{1}{\lambda \left(\|b\|_2 - \frac{1}{\lambda}\right)}\right) = b \tag{2.159}$$

which easily simplifies to

$$x^* = b \left(1 - \frac{1}{\lambda \|b\|_2}\right) \tag{2.160}$$

Now let us consider the case $\lambda \|b\|_2 \leqslant 1$ and assume $x^* \neq 0$. Then we have $q = \frac{x^*}{\|x^*\|_2}$. and the very same computations as in the case $\lambda \|b\|_2 > 1$ lead us to $\|x^*\|_2 = \|b\|_2 - \frac{1}{\lambda}$. However, because of $\lambda \|b\|_2 \leqslant 1$ it follows that we also have $\|b\|_2 - \frac{1}{\lambda} \leqslant 0$ so that the equation $\|x^*\|_2 = \|b\|_2 - \frac{1}{\lambda}$ can only be verified for $\|b\|_2 = \frac{1}{\lambda}$ and $\|x^*\|_2 = 0$. Therefore, our assumption that $x^* \neq 0$ was wrong and as a consequence we must have $x^* = 0$. Combining these the two cases gives us

$$x^* = \text{gshrink}\left(b, \frac{1}{\lambda}\right) \tag{2.161}$$

□

2.5 Summary and concluding remarks

In this chapter we have proven a few important results. For our purposes, the first important result being Proposition 2.17. It guarantees that minimisation strategies cannot get stuck in local minima if we apply them on convex functions.

The other important observation concerns the continuity of convex functions. Theorem 2.29 and Theorem 2.30 show that, under fairly weak assumptions, convex functions are automatically continuous. This result will play a crucial role in later chapters. Also

almost everything about subdifferentials that we have seen here will be important at some point. Especially their existence, as shown in Theorem 2.33, will play a significant role. The two formulas for sums of subdifferentials (Theorem 2.45) and compositions with linear maps (Theorem 2.46) will also be of great use. Finally, one should also mention the two shrinkage operators that we presented. They will play a key role in the split Bregman algorithm which we will analyse in Section 3.2.

3 The Bregman algorithms

Now that we have presented all the necessary preliminary results, we can start to focus our analysis on the Bregman algorithms and their applications. The goal of this chapter will be to present the various forms of Bregman iterative formulations and to discuss their properties and convergence behaviour. We will begin with presenting the standard Bregman algorithm as developed by Osher and his colleagues in [62]. The deduction of the algorithm presented in this thesis will however differ significantly from the deduction given in [62]. There, the authors based their presentation on the fact that a noisy signal S can be decomposed into $S = u + v$, where u is the true signal and v is the noise. They then arrived at the Bregman formulation by trying to recover u with the help of the Rudin-Osher-Fatemi (ROF) model. We will present a deduction that is based on the simple observation that the Bregman algorithm is a combination between a non-orthogonal projection onto a convex set and a regularisation strategy. As a consequence, our interpretation allows us to consider the Bregman framework from a more general point of view than the authors of [62] did. Also, it makes the relation between the original work of Bregman [14] and the algorithm, that we call today Bregman iteration, much more apparent. We will further present an alternative formulation of the Bregman algorithm which has already been discussed in [41, 43]. This alternative formulation is interesting for problems that involve linear operators for which the adjoint is not known explicitly. Beside these "classical" Bregman formulations, we will also consider the split Bregman strategy from Goldstein and Osher [43]. The split Bregman algorithm is a simple trick to reformulate optimisation problems, that could basically not be solved with the Bregman iteration, in such a way that the Bregman framework can still be applied. It will be the basis for our optical flow computations that we will present in Chapter 4. The thorough convergence theory that we will present here is based on the results from [23, 24, 27, 29, 43, 62]. The authors of these references used different vector spaces for their results. Some of the results require Hilbert spaces, whereas other results can be stated in rather general vector spaces only equipped with semi-norms. We will present the results for finite dimensional normed vector spaces. Although this is often much more restrictive than it needs to be, it allows us to present all the results with a common set of requirements, thus making the relations between the different statements a lot clearer. Furthermore, this restriction will be sufficient for our application on optical flow problems later on. Finally, we will shortly present an equivalence between the Bregman algorithm and the augmented Lagrange method. The proof that we will present here is a more detailed reformulation of the proof found in [94]. This equivalence will allow us to interpret the algorithms given in Chapter 4 not only as Bregman iterations, but also as augmented Lagrangian penalty methods.

As in the previous chapter, f will usually denote a convex function from some normed

vector space X to $\mathbb{R}$. Its effective domain, as defined in Definition 2.9, will for simplicity be denoted by Ω.

Before we can formulate the Bregman algorithms, we need to introduce one of the central concepts of the Bregman iteration, namely the Bregman divergence. It has been presented by Bregman in 1967 [14], where it has been used to solve convex optimisation problems through non-orthogonal projections onto convex sets. Our definition of the Bregman divergence will be slightly more general than the one given in [14]. In fact it corresponds to the definition found in [43].

Definition 3.1 (Bregman divergence)
The *Bregman divergence* $D_f^p : \Omega \times \Omega \to \mathbb{R}$ of f is defined as

$$D_f^p(x, y) := f(x) - f(y) - \langle p, x - y \rangle \tag{3.1}$$

where p is a subgradient of f at y.

Note that the Bregman divergence only exists at position (x, y), if there exists a subgradient of f in y. Furthermore, it is not necessarily unique, since a function can have arbitrarily many subgradients at a given location. If X is equal to $\mathbb{R}^n$, then Theorem 2.30 states that convex functions are continuous and Theorem 2.33 implies the existence of at least one subgradient. Furthermore, as presented in Theorem 2.39, the existence of a Gâteaux differential guarantees the uniqueness of the subgradient and as a consequence also of the Bregman divergence.

The Bregman divergence is also known under the name Bregman distance even though it is not a distance function. The Bregman divergence is in general neither symmetric, i.e. $D_f^p(x, y) \neq D_f^p(y, x)$, nor does it satisfy the triangular inequality. Therefore, it cannot be a distance.

The following reflections give a simple analytical interpretation of the Bregman divergence. If we assume that f is a convex and differentiable function on $\mathbb{R}^n$, then its first order Taylor expansion around y is given by

$$f(x) \approx f(y) + \langle \nabla f(y), x - y \rangle \tag{3.2}$$

From this it follows that the difference between $f(x)$ and its Taylor expansion is given by

$$f(x) - f(y) - \langle \nabla f(y), x - y \rangle \tag{3.3}$$

which is exactly the Bregman divergence of f in (x, y). Therefore, the Bregman divergence measures, in a certain sense, how much $f(x)$ differs from its Taylor expansion. For a more geometric interpretation we return to Fig. 2.10. The Bregman divergence of that function is depicted in Fig. 3.1. Based on the figure we can already distinguish one of the useful properties of the Bregman divergence, namely that it is always non-negative. The next example also justifies, in a certain sense, why some authors speak of a Bregman distance. For certain functions, the square root of the Bregman divergence becomes a distance function. A more detailed analysis of this subject can be found in [32, 33]. We will not go further into this direction, as it is of no interest for us at the moment.

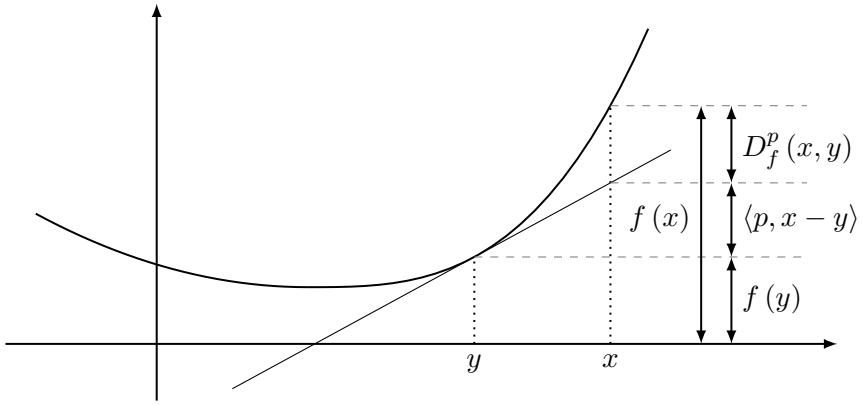


Figure 3.1: Geometric representation of the Bregman divergence D_f^p for a function f with subgradient p at position y.

Example 3.2
The Bregman divergence of $\|\cdot\|_2^2 : \mathbb{R}^n \to \mathbb{R}$ is given by

$$\begin{aligned} D(x,y) &= \|x\|_2^2 - \|y\|_2^2 - \langle 2y, x-y\rangle \\ &= \|x-y\|_2^2 \end{aligned} \tag{3.4}$$

Let us now discuss a few simple but useful characteristics of the Bregman divergence. All theses properties are rather well known and can easily be found in the literature. See for example [16, 62].

Proposition 3.3
Let f and g be two proper convex functions on X. Furthermore, assume that there exist points x, y and z in X at which f, resp. g, possess a subgradient. Then the Bregman divergence satisfies the following properties

1. $D_f^p(x,x) = 0$
2. $D_f^p(x,y) \geqslant 0$
3. The Bregman divergence is convex in its first argument.
4. If f and g are two convex functions and $\lambda > 0$, then we have

$$D_{f+\lambda g}^p(x,y) = D_f^p(x,y) + \lambda D_g^p(x,y) \tag{3.5}$$

5. $D_f^p(x,y) + D_f^{\tilde{p}}(y,z) - D_f^{\tilde{p}}(x,z) = \langle p - \tilde{p}, y - x\rangle$

Proof: The first property follows immediately from the definition of the Bregman divergence. The second property is a clear because of the definition of the subgradient.

In order to show the convexity, we consider $x_1, x_2, y \in \Omega$ and $\lambda \in [0,1]$. Then we have

$$\begin{aligned} &\lambda D_f^p(x_1, y) + (1-\lambda) D_f^p(x_2, y) \\ &\quad = \lambda f(x_1) + (1-\lambda) f(x_2) - f(y) - \langle p, \lambda x_1 + (1-\lambda) x_2 - y \rangle \\ &\quad \geqslant f(\lambda x_1 + (1-\lambda) x_2) - f(y) - \langle p, \lambda x_1 + (1-\lambda) x_2 - y \rangle \\ &\quad = D_f^p(\lambda x_1 + (1-\lambda) x_2, y) \end{aligned} \tag{3.6}$$

The fourth statement is a simple computation. It is enough to expand $D_{f+\lambda g}^p(x, y)$ and to compare it to the definition of the Bregman divergence. Finally, the last claim follows from

$$\begin{aligned} D_f^p(x, y) &= f(x) - f(y) - \langle p, x - y \rangle \\ D_f^{\tilde{p}}(y, z) &= f(y) - f(z) - \langle \tilde{p}, y - z \rangle \\ -D_f^{\tilde{p}}(x, z) &= -f(x) + f(z) + \langle \tilde{p}, x - z \rangle \end{aligned} \tag{3.7}$$

Adding all terms together then gives

$$D_f^p(x, y) + D_f^{\tilde{p}}(y, z) - D_f^{\tilde{p}}(x, z) = \langle p - \tilde{p}, y - x \rangle \tag{3.8}$$

□

Corollary 3.4
If f is strictly convex, then it follows immediately from Proposition 2.41 that we have

1. $D_f^p(x, y) > 0 \quad \forall x \neq y$
2. $D_f^p(x, y) = 0 \Leftrightarrow x = y$

3.1 The standard Bregman iteration

In this section we will see how the Bregman divergence can be used to solve convex programming problems in $\mathbb{R}^n$. The main reason for restricting ourselves to finite dimensional normed vector spaces is that it guarantees us the existence of subgradients for convex functions and allows us to use the shrinkage operators from Section 2.4.

3.1.1 Deduction of the Bregman iteration

Numerous problems in mathematics and physical sciences can be recast in terms of the famous *convex feasibility problem*:

> Given closed convex intersecting sets $C_1, \ldots, C_N$, find a point in $C_1 \cap \ldots \cap C_N$.

Typically, the points in the intersection are the sought-after solutions of a given problem and the sets $C_1, \ldots, C_N$ correspond to some constraints. The convex feasibility problem arises in diverse areas such as best approximation theory, conformal mapping theory,

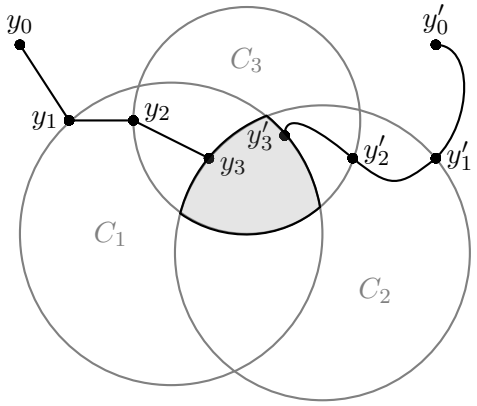


Figure 3.2: The sequence $(y_k)_k$ consists of orthogonal projections onto the sets C_k, whereas the sequence $(y_k')_k$ is made of non-orthogonal projections.

image reconstruction, minimisation of convex functions and statistical estimation. Often, it is possible to calculate the orthogonal projection onto the constraints; thus, denoting the orthogonal projection onto the k^{th} constraint set by P_k, one can solve the convex feasibility problem by the classical method of cyclic orthogonal projections. Given a starting point y_0, generate a sequence $(y_k)_k$ by projecting cyclically onto the constraints, that is

$$y_0 \overset{P_1}{\longmapsto} y_1 \overset{P_2}{\longmapsto} y_2 \overset{P_3}{\longmapsto} \ldots \overset{P_N}{\longmapsto} y_N \overset{P_1}{\longmapsto} y_{N+1} \overset{P_2}{\longmapsto} \ldots$$

The sequence $(y_k)_k$ converges to a solution, if the underlying space is some Euclidean space $\mathbb{R}^l$. This approach was generalised by Bregman in 1967. The novelty in his approach was the use of non-orthogonal projections. He demonstrated that for a "sufficiently well behaved" convex function f, one could also consider the projections with respect to the Bregman divergence D_f, i.e. computing

$$y^{k+1} = \underset{y \in C_k}{\arg\min}\ D_f\left(y, y^k\right) \tag{3.9}$$

Example 3.2 shows us, that using $f(x) = \|x\|_2^2$ would result in orthogonal projections. It follows, that Bregman's approach is really a generalisation of the well known classical method. Figure 3.2 also visualises the idea behind these projections and the difference between an orthogonal and non-orthogonal projection.

Bregman proceeded to show that his novel approach was able to solve convex optimisation problems of the form

$$\underset{x}{\arg\min}\, f(x) \ \text{ such that } Ax = b, \tag{3.10}$$

where f is strictly convex and smooth function, A a matrix and b some vector. He observed that every equation in the linear system corresponds to a closed convex set. Thus, every linear system corresponds to a family of convex sets and solutions of the system always lie in the intersection of these sets. Bregman showed that using his above

mentioned algorithm, he could steer the projections not only towards a solution of the system $Ax = b$, but also to a solution that, at the same time, minimises f. The iterates were basically determined by solving

$$x^{k+1} = \underset{x \in S_k}{\arg\min}\ D_f\left(x, x^k\right) \tag{3.11}$$

where S_k is the set of all the solutions of the k^{th} equation of $Ax = b$. These sets were traversed cyclically. After attaining the last set S_n, one would continue with S_1 again, until the iterates x^k reached a fix point. Bregman proved the convergence of this strategy and demonstrated that these iterates could be computed very efficiently. In practice this algorithm often yields very good convergence speeds. Unfortunately, the algorithm has rather strong requirements. The function f, for example, must be strictly convex and at least continuously differentiable on $\mathbb{R}^n$. This makes it impossible to apply the algorithm on problems of the form

$$\underset{x}{\arg\min} \|x\|_2 \text{ such that } Ax = b \tag{3.12}$$

since $\|\cdot\|_2$ is not differentiable in 0.

Another common approach to solve problems of the form eq. (3.10), is to use regularisation techniques. Here, one approximates the above constrained problem by an unconstrained problem of the form

$$\underset{x}{\arg\min}\ f(x) + \lambda \|Ax - b\|_2^2 \tag{3.13}$$

The idea is, that if λ is large enough, then solutions of eq. (3.13) should be close to solutions of eq. (3.10). This approach is rather popular because unconstrained problems are often simpler to solve. Furthermore, it also allows us to treat problems like eq. (3.12). The drawback of this approach is that one does not know in advance how large λ must be. Therefore, one often considers a sequence $\lambda_0 < \lambda_1 < \ldots < \lambda_n < \ldots$ and computes

$$\underset{x}{\arg\min}\ f(x) + \lambda_k \|Ax - b\|_2^2 \quad \text{for } k = 1, 2, \ldots \tag{3.14}$$

until a fix point is reached. Unfortunately, such algorithms could lead to ill conditioned problems if λ is chosen too large. This would make an efficient numerical solution difficult to realise. Methods that employ strategies with an increasing parameter are also called penalty methods since they penalise deviations from solutions of $Ax = b$.

The idea is now to combine the advantages of the two above mentioned approaches into a single algorithm. Ideally we would like to have a formulation that can handle convex non-differentiable cost functions and that avoids ill conditioned formulations. In a first step, one could for example consider the following iterative strategy:

$$x^{k+1} = \underset{x \in \mathbb{R}^l}{\arg\min}\ D_f\left(x, x^k\right) \text{ such that } Ax = b \tag{3.15}$$

where we require that x solves $Ax = b$ and thus, is not just an element of S_k for some index k, as in the Bregman projection algorithm. However, if the linear system has multiple

solutions or has a very large system matrix, then it might be difficult to determine the iterates x^k. Therefore, we now apply the idea behind eq. (3.13) and obtain

$$x^{k+1} = \arg\min_x \; D_f\left(x, x^k\right) + \lambda \left\|Ax - b\right\|_2^2 \tag{3.16}$$

with some fixed $\lambda > 0$. This iterative strategy motivates the following definition, which coincides with the formulation found in [41, 43, 62, 94]. In [62, 94], the Bregman iteration was formulated as a method for minimising convex functionals of the form $J(u) + H(u)$. However, the convergence theory presented in Section 3.1.2 clearly states that the iterates converge towards the solution of a constrained formulation. Therefore, we define the algorithm, from the beginning on, as method for solving constrained optimisation problems.

Definition 3.5 (Bregman iteration)
Let J and H be two convex functions from $\mathbb{R}^n$ to $\mathbb{R}$ and H a non-negative function with $\min_u H(u) = 0$. Furthermore, assume that $\operatorname{dom} J = \operatorname{dom} H = \mathbb{R}^n$. The *Bregman iteration* of the constrained optimisation problem

$$\arg\min_u J(u) \;\text{ such that }\; H(u) = 0 \tag{3.17}$$

is given by:

1. Choose u^0 arbitrarily, $\lambda > 0$ and $p^0 \in \partial J(u^0)$.
2. Compute iteratively

$$\begin{aligned} u^{k+1} &= \arg\min_u D_J^{p^k}\left(u, u^k\right) + \lambda H(u) \\ &= \arg\min_u J(u) - J\left(u^k\right) - \langle p^k, u - u^k\rangle + \lambda H(u) \\ &= \arg\min_u J(u) - \langle p^k, u - u^k\rangle + \lambda H(u) \end{aligned} \tag{3.18}$$

   where we have $p^k \in \partial J\left(u^k\right) \;\forall k$.

Since J is assumed to be convex and defined on the whole $\mathbb{R}^n$, it follows that it must be continuous and that it has at least one subgradient at every point. Thus p^k always exists, but it is not necessarily unique. Furthermore, note that the function H can also have multiple solutions. The case where H has exactly one solution is not of our interest as it can be reduced to solving a (possibly non-linear) system of equations. Finally, note that the parameter λ inside the Bregman iteration is constant. It can be chosen arbitrarily and does not need to be large at all. It could even be chosen in such a way that it optimises the performance of the algorithms used to solve the subproblem of eq. (3.18). This is certainly a significant advantage over other penalty function methods that require an increasing sequence of λ_k.

At this point we do not know whether the above strategy really works. Showing that the above algorithm really does converge towards a solution of eq. (3.17) will be the task

of Section 3.1.2. The next proposition shows how one can simplify the computation of the subgradients p^k. The proof that we present here is based upon results from [94], where the authors used to Bregman iteration for applications in compressed sensing.

Proposition 3.6
If H is differentiable, the Bregman iteration from Definition 3.5 becomes

$$\begin{aligned} u^{k+1} &= \arg\min_u J(u) - \langle p^k, u - u^k \rangle + \lambda H(u) \\ p^{k+1} &= p^k - \lambda \nabla H\left(u^{k+1}\right) \end{aligned} \tag{3.19}$$

Proof: One only has to show, that $p^{k+1} := p^k - \lambda \nabla H(u^{k+1}) \in \partial J(u^{k+1})$. Since H is convex and differentiable, it follows that $\lambda \nabla H(u^{k+1}) \in \partial \lambda H(u^{k+1})$. Consider now the function $f(x) := -\langle p^k, x - u^k \rangle$. Clearly this function is differentiable and its gradient (and thus also subgradient) is given by $-p^k$. The definition of the iterates u^k implies that u^{k+1} is a minimiser of $J(u) - \langle p^k, u - u^k \rangle + \lambda H(u)$ and from Proposition 2.47 it follows that $0 \in \partial(J(u) - \langle p^k, u - u^k \rangle + \lambda H(u))(u^{k+1})$ must hold. Because of Theorem 2.45 there must exist $p^{k+1} \in \partial J(u^{k+1})$ the fulfills the equation

$$0 = p^{k+1} - p^k + \lambda \nabla H\left(u^{k+1}\right) \tag{3.20}$$

The result now follows by solving this equation for p^{k+1}. □

We see that the equality given by Theorem 2.45 is essential for the Bregman iteration. Although one could basically use any subgradient of J at u^{k+1}, the previous proposition gives us a convenient way of finding a specific one that is easy to obtain. This makes the computation of the iterates much simpler and improves the overall speed of the algorithm. If H were non-differentiable, the same strategy could be applied by replacing $\nabla H(u^{k+1})$ by an arbitrary subgradient of H at u^{k+1}. However, we would have to find such a subgradient at every iteration step, which could prove to be cumbersome. Therefore, it is often desirable to choose H differentiable. For most practical applications this will not be a severe restriction. Before we continue with the theory of the Bregman iteration, let us quickly examine a simple example.

Example 3.7
Consider the following optimisation Problem

$$\arg\min_x \|Ax - b\|_2^2 \quad \text{under the condition} \quad \frac{1}{2}\|Bx - c\|_2^2 = 0 \tag{3.21}$$

where A and B are matrices and b and c two vectors. According to Proposition 3.6, the Bregman iteration of this problem would be

$$\begin{aligned} x^{k+1} &= \arg\min_x \|Ax - b\|_2^2 - \langle p^k, x - x^k \rangle + \frac{\lambda}{2}\|Bx - c\|_2^2 \\ p^{k+1} &= p^k - \lambda B^T\left(Bx^{k+1} - c\right) \end{aligned} \tag{3.22}$$

with x^0 arbitrary and $p^0 = 2A^T(Ax^0 - b)$. Since the cost function above is differentiable, x^{k+1} can be obtained by solving the following linear system

$$\left(A^TA + \lambda B^TB\right)x = A^Tb + \lambda B^Tc + p^k \tag{3.23}$$

At this point a few things should be noted about eq. (3.18). First, it is not clear whether the iterative formulation is simpler to solve than the constrained problem. In general all we know about J is that it is convex, but it could be arbitrarily complicated. Simply taking for J the absolute value already yields problems because it is not differentiable. Even such simple examples as in Example 3.7 could cause difficulties. In that example we had to solve a linear system at each iteration. For large matrices, this certainly becomes a computational burden. Furthermore, this linear system may be ill-conditioned making the efficient determination of a solution even more difficult. On the other hand the performance of such iterative formulations as in eq. (3.18) heavily depends on how fast one can determine the iterates. This issue certainly needs some investigation and will partly be addressed later on. The other important point is of course the convergence, which we will treat in the next section.

3.1.2 Convergence behavior of the standard Bregman algorithm

In this section we will apply the same requirements on J and H as in the previous one. Namely we want J and H to be two convex functions from $\mathbb{R}^n$ to $\mathbb{R}$. Further, H should be a non-negative function with $\min_u H(u) = 0$. Our goal will be to analyse the convergence behaviour of the algorithm given in Definition 3.5 and to show that its iterates converge towards a solution of

$$\arg\min_u J(u) \text{ such that } H(u) = 0 \tag{3.24}$$

In order to proof this assertion we will assume that the Bregman iteration is well defined for J and H. That means we want that

$$\arg\min_u \; D_J^{p^k}\left(u, u^k\right) + \lambda H(u) \tag{3.25}$$

is always solvable. If $J = H = \|\cdot\|_2^2$, then this assumption is true. Thus, there exist functions that verify this hypothesis. But there might very well exist functions for which the minimiser at a certain iteration is infinite. Such functions should be excluded from our reasoning. In the following we will designate the iterates of the Bregman algorithm by u^k, and the subgradients of $\partial J(u^k)$ with p^k. λ will be a strictly positive parameter.

We start with a few elementary results that follow almost immediately from the definition of the Bregman iteration. Propositions 3.8, 3.9, 3.11 and 3.12 and Remark 3.10 are all well known and were already discussed in Osher's original article about the Bregman iteration [62]. The proofs presented here are almost identical to those found in the reference. In order to clarify the arguments, a few details have been added at certain places.

Proposition 3.8
The sequence $\left(H(u^k)\right)_k$ is monotonically decreasing.

$$H\left(u^{k+1}\right) \leqslant H\left(u^k\right) \quad \forall k \in \mathbb{N} \tag{3.26}$$

Proof: By the definition of the iterates, we know that the vector u^{k+1} minimises $D_J^{p^k}\left(u, u^k\right) + \lambda H(u)$ and that $D_J^{p^k}\left(u, u^k\right) \geqslant 0$ for all u. As a consequence, we have

$$\begin{aligned} \lambda H\left(u^{k+1}\right) &\leqslant D_J^{p^k}\left(u^{k+1}, u^k\right) + \lambda H\left(u^{k+1}\right) \\ &\leqslant D_J^{p^k}\left(u^k, u^k\right) + \lambda H\left(u^k\right) \\ &= \lambda H\left(u^k\right) \end{aligned} \tag{3.27}$$

Dividing by $\lambda > 0$ gives the desired result. □

Proposition 3.9
Let $u \in \mathbb{R}^n$ and assume that H is differentiable. Then we have for all $\lambda > 0$

$$D_J^{p^k}\left(u, u^k\right) + D_J^{p^{k-1}}\left(u^k, u^{k-1}\right) + \lambda H\left(u^k\right) \leqslant \lambda H(u) + D_J^{p^{k-1}}\left(u, u^{k-1}\right) \tag{3.28}$$

Proof: By using Proposition 3.3 with $p = p^k$ and $\tilde{p} = p^{k-1}$ and by the fact that $p^k - p^{k-1} = -\lambda \nabla H\left(u^k\right)$ we get

$$\begin{aligned} D_J^{p^k}\left(u, u^k\right) + D_J^{p^{k-1}}\left(u^k, u^{k-1}\right) - D_J^{p^{k-1}}\left(u, u^{k-1}\right) &= -\lambda \langle \nabla H\left(u^k\right), u^k - u \rangle \\ &= \lambda \langle \nabla H\left(u^k\right), u - u^k \rangle \\ &\leqslant \lambda H(u) - \lambda H\left(u^k\right) \end{aligned}$$

where the last inequality follows from the fact that $\nabla H\left(u^k\right)$ is a subgradient of H at position u^k. □

Remark 3.10
From the previous proposition we obtain, for the particular choice $u = \tilde{u}$, where $\tilde{u}$ is a solution of H (i.e. $H(\tilde{u}) = 0$), the following result

$$\begin{aligned} D_J^{p^k}\left(\tilde{u}, u^k\right) &\leqslant D_J^{p^k}\left(\tilde{u}, u^k\right) + D_J^{p^{k-1}}\left(u^k, u^{k-1}\right) \\ &\leqslant D_J^{p^k}\left(\tilde{u}, u^k\right) + D_J^{p^{k-1}}\left(u^k, u^{k-1}\right) + \lambda H\left(u^k\right) - \lambda H(\tilde{u}) \\ &\leqslant D_J^{p^{k-1}}\left(\tilde{u}, u^{k-1}\right) \end{aligned} \tag{3.29}$$

where we used the inequality from the previous proposition in eq. (3.29).

The preceding observations provide a few interesting results. We know that the Bregman divergence is always non-negative and if J is strictly convex, then we have $D_J^p(x,y) = 0$ if and only if $x = y$. It follows, that for such J, the sequence $(H(u^k))_k$, presented in Proposition 3.8, is even strictly monotonically decreasing as long as $u^{k+1} \neq u^k$. Furthermore, it implies that

$$0 \leqslant D_J^{p^k}\left(\tilde{u}, u^k\right) < D_J^{p^{k-1}}\left(\tilde{u}, u^{k-1}\right) \tag{3.30}$$

holds in Remark 3.10. Thus, we can conclude that for strictly convex J, the iterates converge towards a solution of H. The next proposition will give us an estimate how fast this convergence is and prove that the strict convexity is in fact not necessary.

Proposition 3.11
If $\tilde{u}$ is a solution of H and $D_J^{p^0}(\tilde{u}, u^0) < +\infty$, then we have for all $\lambda > 0$

$$0 = H(\tilde{u}) \leqslant H\left(u^k\right) \leqslant \frac{D_J^{p^0}(\tilde{u}, u^0)}{\lambda k} \tag{3.31}$$

Thus, the iterates u^k always converge towards a solution of H.

Proof: Because of eq. (3.28) we have for all $i \geqslant 1$

$$D_J^{p^i}\left(u^i, u^{i-1}\right) + \lambda H\left(u^i\right) - \underbrace{\lambda H(\tilde{u})}_{=0} \leqslant D_J^{p^{i-1}}\left(\tilde{u}, u^{i-1}\right) - D_J^{p^i}\left(\tilde{u}, u^i\right) \tag{3.32}$$

$$\Leftrightarrow D_J^{p^i}\left(u^i, u^{i-1}\right) + \lambda H\left(u^i\right) \leqslant D_J^{p^{i-1}}\left(\tilde{u}, u^{i-1}\right) - D_J^{p^i}\left(\tilde{u}, u^i\right) \tag{3.33}$$

and by summing both sides for i from 1 to k we receive

$$\sum_{i=1}^{k}\left[D_J^{p^i}\left(u^i, u^{i-1}\right) + \lambda H\left(u^i\right)\right] \leqslant D_J^{p^0}\left(\tilde{u}, u^0\right) - D_J^{p^k}\left(\tilde{u}, u^k\right) \tag{3.34}$$

But this implies that

$$D_J^{p^k}\left(\tilde{u}, u^k\right) + \sum_{i=1}^{k}\left[D_J^{p^{i-1}}\left(u^i, u^{i-1}\right) + \lambda H\left(u^i\right)\right] \leqslant D_J^{p^0}\left(\tilde{u}, u^0\right) \tag{3.35}$$

Proposition 3.8 now says that $H(u^k) \leqslant H(u^i)$ for all $i = 0, \ldots, k$ and since we also have $D_J^{p^{i-1}}(u^i, u^{i-1}) \geqslant 0$, we may further conclude that

$$k\lambda H\left(u^k\right) \leqslant \sum_{i=1}^{k}\left[D_J^{p^{i-1}}\left(u^i, u^{i-1}\right) + \lambda H\left(u^i\right)\right] \tag{3.36}$$

Inserting the last inequality into the second last one results in

$$D_J^{p^k}\left(\tilde{u}, u^k\right) + k\lambda H\left(u^k\right) \leqslant D_J^{p^0}\left(\tilde{u}, u^0\right) \tag{3.37}$$

Since the Bregman divergence is non-negative, we immediately obtain

$$0 = H(\tilde{u}) \leqslant H\left(u^k\right) \leqslant \frac{D_J^{p^0}(\tilde{u}, u^0)}{\lambda k} \tag{3.38}$$

□

Let us now shortly consider the case where our constraint H depends on some data f. Assume that we do not know f exactly, but only a noisy version which we will call g. Furthermore, assume that $\tilde{u}$ is a solution of $H(\cdot, f)$ that minimizes our cost function J and that $H(\tilde{u}, g) \leqslant \delta^2$. For some $\delta^2 > 0$. Then the following proposition gives us information about the convergence when we have such noisy data.

Proposition 3.12
Assume f, g and $\tilde{u}$ are as described above. Then the Bregman divergence between $\tilde{u}$ and u^k is decreasing as long as $H(u^k, g) > \delta^2$; more precisely,

$$D_J^{p^k}\left(\tilde{u}, u^k\right) < D_J^{p^{k-1}}\left(\tilde{u}, u^{k-1}\right)$$

Proof: From eq. (3.28) we know that we have

$$\begin{aligned} D_J^{p^k}\left(\tilde{u}, u^k\right) + D_J^{p^{k-1}}\left(u^k, u^{k-1}\right) + \lambda H\left(u^k, g\right) &\leqslant \lambda H(\tilde{u}, g) + D_J^{p^{k-1}}\left(\tilde{u}, u^{k-1}\right) \\ &\leqslant \lambda\delta^2 + D_J^{p^{k-1}}\left(\tilde{u}, u^{k-1}\right) \end{aligned} \tag{3.39}$$

The result follows now immediately from the fact that $D_J^{p^{k-1}}\left(u^k, u^{k-1}\right) \geqslant 0$ and that $\lambda H\left(u^k, g\right) > \lambda\delta^2$. □

This implies that the Bregman iteration remains stable even if our data contains noise. For practical applications this result is quite important since the assumption of noise free data is rather unrealistic. Also note that the above assumption is only useful if δ^2 is small.

So far we have seen that the iterates converge towards a solution of H. But at this point we do not know whether this solution also minimises our cost function J. If H has a unique solution, than the above theory is sufficient. Thus it seems that such iterates might be good candidates for solving eq. (3.17). The following proposition now states, that under certain assumptions, the iterates that solve H also minimise our cost function, even if H has multiple solutions. This highly important result was first pointed out in [94], where the authors analysed the convergence behavior of the Bregman iteration applied to the basis pursuit problem.

Proposition 3.13
Assume there exists a u^0 such that it is possible to choose $p^0 = 0$ in eq. (3.19). Furthermore, assume that

$$H(u) = h(Au - b) \tag{3.40}$$

where A is some matrix, b an arbitrary vector and h is a differentiable non-negative convex function that only vanishes at 0. If an iterate u^k fulfills $H(u^k) = 0$, i.e. it solves $Au = b$, then that iterate is also a solution of the constrained optimisation problem of eq. (3.17).

PROOF: Since h (and thus also H) is differentiable, we have

$$\nabla H(u) = A^T \nabla h(Au - b) \tag{3.41}$$

Because of Proposition 3.6 we have

$$\begin{aligned} p^k &= p^{k-1} - \lambda \nabla H\left(u^k\right) \\ &= p^{k-2} - \lambda \nabla H\left(u^{k-1}\right) - \lambda \nabla H\left(u^k\right) \\ &= p^0 - \lambda \sum_{i=1}^{k} \nabla H\left(u^i\right) \\ &= -\lambda \sum_{i=1}^{k} \nabla H\left(u^i\right) \\ &= -\lambda A^T \sum_{i=1}^{k} \nabla h\left(u^i\right) \end{aligned} \tag{3.42}$$

From the definition of the subgradient for J at u^k, it follows that

$$\begin{aligned} J\left(u^k\right) &\leqslant J(u) - \langle p^k, u - u^k \rangle \\ &= J(u) + \langle A^T \sum_{i=1}^{k} \nabla h\left(Au^i - b\right), u - u^k \rangle \\ &= J(u) + \langle \sum_{i=1}^{k} \nabla h\left(Au^i - b\right), Au - Au^k \rangle \\ &= J(u) + \langle \sum_{i=1}^{k} \nabla h\left(Au^i - b\right), Au - b \rangle \end{aligned} \tag{3.43}$$

This holds for all u. Especially for all solutions $\tilde{u}$ of eq. (3.17). Plugging in any such solution immediately gives $J(u^k) \leqslant J(\tilde{u})$ and thus u^k is also a solution of eq. (3.17). □

Remark 3.14
The previous proposition requires that h vanishes only at 0. However, the linear system $Au = b$ can have arbitrary many solutions. Thus H can have multiple solutions too. The requirement the h only vanishes at 0 is essential in the previous proof. It enforces that every zero of H solves the linear system. Without this information, the above proof would not be valid.

At this point we know the following things:

1. The iterates of our Bregman algorithm converge towards a solution of H (Proposition 3.11).
2. If an iterate u^k fulfills $H(u^k) = 0$ and if H is of a certain special form, then that iterate also solves eq. (3.17) (Proposition 3.13).

Thus we have guaranteed convergence under a certain number of restrictions. For our optical flow problems that we will consider later, these restrictions will fit naturally into our modelling and do not cause any problems for us.

Basically, the previous statements also provide some kind of stopping criterion for the Bregman iteration. By evaluating $H(u^k)$ after each iteration, one could simply stop as soon as this value drops below a certain threshold. However, this approach might be a bit unreliable since we do not know how far we are away from the real solution, even if $H(u^k)$ is very small.

During the rest of this chapter we will focus on the special case that $h(x) := \|x\|_2^2$. In that case, it is possible to make some kind of estimate for the error at each iteration step. Thus, from now on, we assume that A is a given $m \times n$ matrix and b a known vector in $\mathbb{R}^n$. The problem that we consider is

$$\arg\min_u \; J(u) \quad \text{such that} \quad \frac{1}{2}\|Au - b\|_2^2 = 0 \tag{3.44}$$

Proposition 3.6 implies that we have the following algorithm

$$\begin{aligned} u^{k+1} &= \arg\min_u J(u) - \langle p^k, u - u^k \rangle + \frac{\lambda}{2}\|Au - b\|_2^2 \\ p^{k+1} &= p^k + \lambda A^T\left(b - Au^{k+1}\right) \end{aligned} \tag{3.45}$$

For technical reasons we will continue to assume that it is possible to choose u^0 such that $p^0 = 0$ can be used. This can always be done as long as J has a minimum at some finite point.

Definition 3.15 (Minimising Solution)
A vector $\tilde{u}$ is called J *minimising solution* of $Au = b$, if $A\tilde{u} = b$ and $J(\tilde{u}) \leqslant J(v)$ for all other v that fulfill $Av = b$.

Definition 3.16 (Source Condition)
Let $\tilde{u}$ be a J minimising solution of $Au = b$. We say $\tilde{u}$ satisfies the *source condition* if there exists an ω such that $A^T\omega \in \partial J(\tilde{u})$.

The source condition can, in a certain sense, be interpreted as an additional regularity condition that we impose on the solution. Not only do we require that the minimising solution has subgradient, we even want that there exists a subgradient that lies in the range of A^T. Requirements like the source condition are a frequent tool in the analysis of inverse problems.

The next theorem shows that it is possible to give an estimate for the error if this source condition holds. This result was also presented in [27], where the authors discussed the convergence behavior of the Bregman iteration in the context of inverse scale space methods for image restoration purposes. The proof that we present here is essentially the same as in the reference.

Theorem 3.17
Let $\tilde{u}$ be a J minimising solution of $Au = b$ and assume that the source condition holds, i.e. there exists a vector $\xi \in \partial J(\tilde{u})$ such that $\xi = A^T q$ for some vector q. Furthermore,

assume that it is possible to choose u^0 such that $p^0 = 0$ is a subgradient of J at u^0. Then we have the following estimation for the iterates u^k of eq. (3.45).

$$D_J^{p^k}\left(\tilde{u}, u^k\right) \leqslant \frac{\|q\|_2^2}{2\lambda k} \quad \forall k \in \mathbb{N}^* \tag{3.46}$$

Proof: As we know from Proposition 3.6, the necessary subgradients at each step of the Bregman iteration can be determined by

$$\begin{aligned} p^k &= p^{k-1} + \lambda A^T\left(b - Au^k\right) \\ &= p^0 + \sum_{j=1}^{k} \lambda A^T\left(b - Au^j\right) \\ &= \sum_{j=1}^{k} \lambda A^T\left(b - Au^j\right) \end{aligned} \tag{3.47}$$

We now define

$$\begin{aligned} x^k &= \lambda \sum_{j=1}^{k}\left(b - Au^j\right) \quad \text{for } k \geqslant 1 \\ x^0 &= 0 \end{aligned} \tag{3.48}$$

Then we have the following identities

$$\begin{aligned} p^k &= A^T x^k \\ x^{k-1} - x^k &= \lambda \sum_{j=1}^{k-1}\left(b - Au^j\right) - \lambda \sum_{j=1}^{k}\left(b - Au^j\right) = \lambda\left(Au^k - b\right) \end{aligned} \tag{3.49}$$

Let us now fix $k \in \mathbb{N}^*$, then for any $j \in \mathbb{N}^*$ with $j \leqslant k$ we have

$$\begin{aligned} \lambda D_J^{p^j}\left(\tilde{u}, u^j\right) + \lambda D_J^{\xi}\left(u^j, \tilde{u}\right) &= \lambda\left(-\langle p^j, \tilde{u} - u^j\rangle - \langle \xi, u^j - \tilde{u}\rangle\right) \\ &= \lambda\langle p^j - \xi, u^j - \tilde{u}\rangle \\ &= \langle A^T x^j - A^T q, \lambda\left(u^j - \tilde{u}\right)\rangle \\ &= \langle x^j - q, \lambda\left(Au^j - A\tilde{u}\right)\rangle \\ &= \langle x^j - q, \lambda\left(Au^j - b\right)\rangle \\ &= \langle x^j - q, x^{j-1} - x^j\rangle \\ &= \frac{1}{4}\left(\left\|x^{j-1} - q\right\|_2^2 - \left\|\left(x^j - q\right) - \left(x^{j-1} - x^j\right)\right\|_2^2\right) \end{aligned} \tag{3.50}$$

where eq. (3.50) follows from the identity

$$\langle z_1, z_2\rangle = \frac{1}{4}\left(\|z_1 + z_2\|_2^2 - \|z_1 - z_2\|_2^2\right) \tag{3.51}$$

with $z_1 := x^j - q$ and $z_2 := x^{j-1} - x^j$. This identity holds in any real-valued Hilbert space. It can be easily verified by multiplying out the right hand side. Thus, we now have

$$\lambda D_J^{p^j}\left(\tilde{u}, u^j\right) + \lambda D_J^{\xi}\left(u^j, \tilde{u}\right) = \frac{1}{4}\left(\|z_1 + z_2\|_2^2 - \|z_1 - z_2\|_2^2\right) \tag{3.52}$$

By expanding the lefthand side, it is also easy to verify that the following identity

$$\|z_1 + z_2\|_2^2 + \|z_1 - z_2\|_2^2 = 2\left(\|z_1\|_2^2 + \|z_2\|_2^2\right) \tag{3.53}$$

holds in any Hilbert space. But this implies that we also have

$$\|z_1 - z_2\|_2^2 = 2\left(\|z_1\|_2^2 + \|z_2\|_2^2\right) - \|z_1 + z_2\|_2^2 \tag{3.54}$$

Finally, from this identity we can conclude that

$$\begin{aligned}
\lambda D_J^{p^j}\left(\tilde{u}, u^j\right) + \lambda D_J^{\xi}\left(u^j, \tilde{u}\right) &= \frac{1}{4}\left(\|z_1 + z_2\|_2^2 - \|z_1 - z_2\|_2^2\right) \\
&= \frac{1}{2}\left(\|z_1 + z_2\|_2^2 - \|z_1\|_2^2 - \|z_2\|_2^2\right) \\
&= \frac{1}{2}\left(\left\|x^{j-1} - q\right\|_2^2 - \left\|x^j - q\right\|_2^2 - \left\|x^{j-1} - x^j\right\|_2^2\right) \\
&\leqslant \frac{1}{2}\left\|x^{j-1} - q\right\|_2^2 - \frac{1}{2}\left\|x^j - q\right\|_2^2
\end{aligned} \tag{3.55}$$

Using the previous estimation, it follows that

$$\begin{aligned}
\sum_{j=1}^{k}\left(\lambda D_J^{p^j}\left(\tilde{u}, u^j\right) + \lambda D_J^{\xi}\left(u^j, \tilde{u}\right)\right) &\leqslant \frac{1}{2}\sum_{j=1}^{k}\left(\left\|x^{j-1} - q\right\|_2^2 - \left\|x^j - q\right\|_2^2\right) \\
&\leqslant \frac{1}{2}\left(\left\|x^0 - q\right\|_2^2 - \left\|x^k - q\right\|_2^2\right)
\end{aligned} \tag{3.56}$$

Finally, because of Remark 3.10 we have for all $j \leqslant k$

$$D_J^{p^k}\left(\tilde{u}, u^k\right) \leqslant D_J^{p^j}\left(\tilde{u}, u^j\right) \tag{3.57}$$

and by summing up both sides for j from 1 to k we obtain

$$kD_J^{p^k}\left(\tilde{u}, u^k\right) \leqslant \sum_{j=1}^{k} D_J^{p^j}\left(\tilde{u}, u^j\right) \leqslant \sum_{j=1}^{k}\left(D_J^{p^j}\left(\tilde{u}, u^j\right) + D_J^{\xi}\left(u^j, \tilde{u}\right)\right) \tag{3.58}$$

where the second step follows from $0 \leqslant D_J^{\xi}\left(u^j, \tilde{u}\right)$. Equation (3.56) leads us now to the desired estimation

$$D_J^{p^k}\left(\tilde{u}, u^k\right) \leqslant \frac{1}{2\lambda k}\left(\left\|x^0 - q\right\|_2^2 - \left\|x^k - q\right\|_2^2\right) \leqslant \frac{1}{2\lambda k}\left\|x^0 - q\right\|_2^2 = \frac{1}{2\lambda k}\|q\|_2^2 \tag{3.59}$$

□

Remark 3.18
Proposition 3.3 says that the Bregman divergence is always non-negative. If furthermore, J is strictly convex, then Corollary 3.4 claims that we also have $D_J^{p^k}(\tilde{u}, u^k) > 0$ as long as $\tilde{u} \neq u^k$. It follows that $D_J^{p^k}(\tilde{u}, u^k) = 0 \Leftrightarrow \tilde{u} = u^k$. In this setting the above theorem guarantees that the iterates converge towards the unique solution $\tilde{u}$. Note that if J is not strictly convex, then $D_J^{p^k}(\tilde{u}, u^k) = 0$ can hold even if $\tilde{u} \neq u^k$. Consider for example $J(u) = |u|$ and $\tilde{u} = 0$, $u^k = 1$ and $p^k = 1$. Then we have $D_J^{p^k}(\tilde{u}, u^k) = 0$ but $\tilde{u} \neq u^k$.

3.1.3 Alternative formulation of the standard Bregman algorithm

In this section we will show one possible approach to simplify the computations in the Bregman algorithm. We will continue to consider the following problem

$$\arg\min_u \; J(u) \quad \text{such that} \quad \frac{1}{2}\|Au - b\|_2^2 = 0 \tag{3.60}$$

where J is a convex function from $\mathbb{R}^n$ to $\mathbb{R}$. A should be a $m \times n$ matrix and b a vector in $\mathbb{R}^m$. We also assume that the linear system $Au = b$ has at least one solution. Because of Definition 3.5 and Proposition 3.6, the Bregman iterations for eq. (3.60) are given by

$$\begin{aligned} u^{k+1} &= \arg\min_u D_J^{p^k}\left(u, u^k\right) + \frac{\lambda}{2}\|Au - b\|_2^2 \\ &= \arg\min_u J(u) - \langle p^k, u\rangle + \frac{\lambda}{2}\|Au - b\|_2^2 \\ p^{k+1} &= p^k - \lambda A^T\left(Au^{k+1} - b\right) \end{aligned} \tag{3.61}$$

Let us make again the assumption that it is possible to find a u^0 such that we can set $p^0 = 0$. As we have already seen in the proof of Proposition 3.13, the variables p^k can then be given by

$$\begin{aligned} p^k &= p^{k-1} - \lambda A^T\left(Au^k - b\right) \\ &= p^0 - \lambda\sum_{j=1}^{k} A^T\left(Au^j - b\right) \\ &= \lambda\sum_{j=1}^{k} A^T\left(b - Au^j\right) \end{aligned} \tag{3.62}$$

Now let us have a look at the cost function from eq. (3.61). The last two terms can be reformulated in then following way

$$\begin{aligned} \frac{\lambda}{2}\|Au - b\|_2^2 - \langle p^k, u\rangle &= \frac{\lambda}{2}\|Au - b\|_2^2 - \lambda\langle\sum_{j=1}^{k} A^T\left(b - Au^j\right), u\rangle \\ &= \frac{\lambda}{2}\|Au\|_2^2 + \frac{\lambda}{2}\|b\|_2^2 - \lambda\langle Au, b\rangle - \lambda\langle\sum_{j=1}^{k} A^T\left(b - Au^j\right), u\rangle \end{aligned}$$

$$= \frac{\lambda}{2}\|Au\|_2^2 + \frac{\lambda}{2}\|b\|_2^2 - \lambda\langle b + \sum_{j=1}^{k}\left(b - Au^j\right), Au\rangle \tag{3.63}$$

If we define $b^k := b + \sum_{j=1}^{k}\left(b - Au^j\right)$, then eq. (3.63) can be rewritten as

$$\frac{\lambda}{2}\|Au - b\|_2^2 - \langle p^k, u\rangle = \frac{\lambda}{2}\|Au\|_2^2 + \frac{\lambda}{2}\|b\|_2^2 - \lambda\langle b^k, Au\rangle \tag{3.64}$$

and the righthand side of this equation can be reformulated as

$$\begin{aligned}\frac{\lambda}{2}\|Au\|_2^2 + \frac{\lambda}{2}\|b\|_2^2 - \lambda\langle b^k, Au\rangle &= \frac{\lambda}{2}\|Au\|_2^2 + \frac{\lambda}{2}\|b\|_2^2 - \lambda\langle b^k, Au\rangle + \frac{\lambda}{2}\left\|b^k\right\|_2^2 - \frac{\lambda}{2}\left\|b^k\right\|_2^2 \\ &= \frac{\lambda}{2}\left\|Au - b^k\right\|_2^2 + \frac{\lambda}{2}\|b\|_2^2 - \frac{\lambda}{2}\left\|b^k\right\|_2^2 \end{aligned} \tag{3.65}$$

In other words, the functions $J(u) - \langle p^k, u\rangle + \frac{\lambda}{2}\|Au - b\|_2^2$ and $J(u) + \frac{\lambda}{2}\|Au - b^k\|_2^2$ are minimized by the same u, since they only differ by a constant independent of u. This motivates the following proposition. Its result can also be found in [94], however, we will not present the proof found within this reference, but use the slightly simpler proof given in [41].

Proposition 3.19 (Alternative form of the Bregman iteration)
The Bregman iteration defined in Definition 3.5 and Proposition 3.6 for solving the constrained optimisation problem

$$\arg\min_u J(u) \quad \text{such that} \quad \frac{1}{2}\|Au - b\|_2^2 = 0 \tag{3.66}$$

can be simplified to the following iterative form

$$\begin{aligned} b^0 &= b \\ u^{k+1} &= \arg\min_u \; J(u) + \frac{\lambda}{2}\left\|Au - b^k\right\|_2^2 \\ b^{k+1} &= b^k + b - Au^{k+1} \end{aligned} \tag{3.67}$$

if we can choose a u^0 such that $p^0 = 0$.

Proof: As seen above we have

$$J(u) - \langle p^k, u\rangle + \frac{\lambda}{2}\|Au - b\|_2^2 = J(u) + \frac{\lambda}{2}\left\|Au - b^k\right\|_2^2 + \frac{\lambda}{2}\|b\|_2^2 - \frac{\lambda}{2}\left\|b^k\right\|_2^2 \tag{3.68}$$

Since $\frac{\lambda}{2}\|b\|_2^2 - \frac{\lambda}{2}\left\|b^k\right\|_2^2$ is constant in u, both objective functions are minimized by the same u at every iteration step k. □

The advantage of this alternative formulation is that we do not explicitly need to know the matrix A^T. This can be of advantage when the matrix A is only available through some approximation. In that case the transpose is often not known. For example, assume

that A were the Jacobian matrix of some complicated function F at a certain point z. Then the action of this Jacobian on an arbitrary vector v can be approximated by the simple difference formula

$$Av \approx \frac{F(z+\varepsilon v) - F(z)}{\varepsilon} \tag{3.69}$$

with $\varepsilon > 0$. Unfortunately there is no such formula to compute $A^T v$. Thus, it is possible to have cases where minimising this new formulation might be much easier to perform than the standard Bregman iteration. Because of the equivalence of the two formulations the iterates, given by this new Bregman algorithm, have the same properties as the ones of the standard Bregman iteration. Thus, all the convergence results that we found in the previous section also apply in this case. Interestingly this new formulation even allows a somewhat simpler proof for Proposition 3.13. It was originally discovered by Goldstein and Osher and presented in [43].

Proposition 3.20
An iterate u^k of the algorithm presented in eq. (3.67) which satisfies $Au^k = b$ is also a solution to the original constrained problem from eq. (3.60).

Proof: Let u^k and b^k be such that $Au^k = b$ and

$$u^k = \arg\min_u J(u) + \frac{\lambda}{2}\left\|Au - b^k\right\|_2^2 \tag{3.70}$$

Let $\tilde{u}$ be a true solution of eq. (3.60). Then $A\tilde{u} = b = Au^k$ and therefore,

$$\left\|Au^k - b^k\right\|_2^2 = \left\|A\tilde{u} - b^k\right\|_2^2 \tag{3.71}$$

Because u^k satisfies eq. (3.70) we have

$$J\left(u^k\right) + \frac{\lambda}{2}\left\|Au^k - b^k\right\|_2^2 \leqslant J(\tilde{u}) + \frac{\lambda}{2}\left\|A\tilde{u} - b^k\right\|_2^2 \tag{3.72}$$

Now eq. (3.71) implies that

$$J\left(u^k\right) \leqslant J(\tilde{u}) \tag{3.73}$$

must hold. Since $\tilde{u}$ satisfies the original problem, this inequality can be sharpened to an equality, showing that u^k solves eq. (3.60). □

Example 3.21
Let us consider once again Example 3.7, but this time we will solve the problem with the alternative Bregman formulation. The iterates are computed as follows

$$\begin{aligned} x^{k+1} &= \arg\min_x \|Ax - b\|_2^2 + \frac{\lambda}{2}\left\|Bx - c^k\right\|_2^2 \\ c^{k+1} &= c^k + c - Bx^{k+1} \end{aligned} \tag{3.74}$$

with $x^0 = 0$ and $c^0 = c$. Interestingly the requirement that there must be an x^0 such that $p^0 = 0$ is a subgradient is only needed for the deduction of the algorithm. x^0 is nowhere

used in the iterations. Therefore, we can simply set $x^0 = 0$. Similarly as in Example 3.7, the iterates can again be determined by solving

$$\left(A^T A + \lambda B^T B\right) x = A^T b + \lambda B^T c^k \tag{3.75}$$

Note in this case the standard Bregman iteration and its alternative form are almost identical. Both suffer under same potential problems that we already mentioned in Example 3.7.

3.1.4 The Bregman iteration and the Lagrangian penalty method

In this section we will show that the Bregman algorithm can be interpreted as an augmented Lagrangian penalty method. This equivalence has already been pointed out by Yin et al. in [94]. The reason why we discuss this result here is that it illustrates the relationship between the algorithms which we will present in Chapter 4 and other well known approaches. In the following we assume that J is a smooth and convex function from $\mathbb{R}^n$ to $\mathbb{R}$, $A \in \mathbb{R}^{m \times n}$ a matrix and $b \in \mathbb{R}^m$ a vector. Let us consider the following optimisation problem

$$\arg\min_x \; J(x) \quad \text{such that} \quad Ax - b = 0 \tag{3.76}$$

where we assume that the linear system $Ax = b$ has at least one solution. In this context let us quickly recall a few well known definitions. They can also be found in [40], where they have been used to present a general approach to optimisation algorithms.

Definition 3.22 (Lagrange function)
The function $L : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ defined by

$$L(x, \mu) = J(x) + \langle \mu, Ax - b \rangle \tag{3.77}$$

is called the *Lagrange function (Lagrangian)* corresponding to eq. (3.76).

Definition 3.23 (Karush-Kuhn-Tucker point)
The conditions

$$\begin{aligned} \nabla_x L(x, \mu) &= 0 \\ Ax - b &= 0 \end{aligned} \tag{3.78}$$

where we define

$$\nabla_x L(x, \mu) := \nabla J(x) + A^T \mu \tag{3.79}$$

are called the *Karush-Kuhn-Tucker conditions* of the constrained optimisation problem from eq. (3.76). Any vector $(x^*, \mu^*) \in \mathbb{R}^n \times \mathbb{R}^m$ that fulfills these conditions is called a *Karush-Kuhn-Tucker point* or sometimes also a *KKT point*.

The Lagrange function and the Karush-Kuhn-Tucker conditions are classical tools that are used to state when solutions of optimisation problems exist. They can be formulated in much more general framework than we do it here. See for example [17]. Let us now assume that x^* is a solution of eq. (3.76). Then x^* is obviously also a solution of

$$\arg\min_x \; J(x) + \frac{\alpha}{2} \|Ax - b\|_2^2 \quad \text{such that} \quad Ax - b = 0 \tag{3.80}$$

with $\alpha > 0$. The Lagrangian of eq. (3.80) is given by

$$L(x, \mu, \alpha) := J(x) + \frac{\alpha}{2} \|Ax - b\|_2^2 + \langle \mu, Ax - b \rangle \tag{3.81}$$

and is called the *augmented Lagrangian* of eq. (3.76). This augmented Lagrangian can be used to formulate an algorithm to solve the initial problem given in eq. (3.76). It is presented in Algorithm 3.1 and is also known as the *augmented Lagrangian penalty method*. It is a relatively popular method to solve constrained optimisation problems

Data: J, A, b
Result: x^* minimising eq. (3.76)
Initialize: $x^0 \in \mathbb{R}^n$, $\mu^0 \in \mathbb{R}^m$, $\alpha_0 > 0$, $c \in (0, 1]$
while x^k *is not a KKT point of eq.* (3.76) **do**
  $x^{k+1} \leftarrow \arg\min_x L\left(x, \mu^k, \alpha_k\right)$
  $\mu^{k+1} \leftarrow \mu^k + \alpha_k \left(Ax^{k+1} - b\right)$
  if $\left\|Ax^{k+1} - b\right\|_2^2 \geqslant c \left\|Ax^k - b\right\|_2^2$ **then**
    $\alpha_{k+1} = 10\alpha_k$
  else
    $\alpha_{k+1} = \alpha_k$
  end
  $k = k + 1$
end

Algorithm 3.1: The augmented Lagrangian penalty method for solving eq. (3.76).

such as eq. (3.76). A more detailed analysis of this algorithm can also be found in [40]. The algorithm is for us only interesting insofar that it can be linked to the Bregman iteration. In order to see this, let us now slightly reformulate the different steps of Algorithm 3.1.

$$\begin{aligned} \arg\min_x \; L\left(x, \mu^k, \alpha_k\right) &= \arg\min_x \; J(x) + \frac{\alpha_k}{2} \|Ax - b\|_2^2 + \langle \mu^k, Ax - b \rangle \\ &= \arg\min_x \; J(x) - J\left(x^k\right) + \langle \mu^k, Ax - b \rangle - \langle \mu^k, Ax^k \rangle + \frac{\alpha_k}{2} \|Ax - b\|_2^2 \\ &= \arg\min_x \; J(x) - J\left(x^k\right) + \langle \mu^k, Ax - Ax^k \rangle - \langle \mu^k, b \rangle + \frac{\alpha_k}{2} \|Ax - b\|_2^2 \end{aligned} \tag{3.82}$$

$$= \arg\min_x \; J(x) - J\left(x^k\right) + \langle A^T\mu^k, x - x^k\rangle - \langle \mu^k, b\rangle + \frac{\alpha_k}{2}\|Ax-b\|_2^2$$
$$= \arg\min_x \; J(x) - J\left(x^k\right) - \langle -A^T\mu^k, x - x^k\rangle + \frac{\alpha_k}{2}\|Ax-b\|_2^2$$

Now let us define $p^k := -A^T\mu^k$. Then the above formulation becomes

$$\begin{aligned} x^{k+1} &= \arg\min_x J(x) - J\left(x^k\right) - \langle p^k, x - x^k\rangle + \frac{\alpha_k}{2}\|Ax-b\|_2^2 \\ p^{k+1} &= p^k - \alpha_k A^T\left(Ax^{k+1} - b\right) \end{aligned} \tag{3.83}$$

This looks almost like our Bregman algorithm from Proposition 3.6. Actually if $\mu^0 = 0$ and if $0 = -A^T\mu^0 = p^0 \in \partial J(u^0)$ and $c = 1$ in Algorithm 3.1, then Proposition 3.8 guarantees that $\|Ax^{k+1} - b\|_2^2 \leqslant \|Ax^k - b\|_2^2$ for all k and thus α_k is never increased and remains constant. Under these conditions, the Bregman iteration applied to eq. (3.76) is equivalent to the augmented Lagrangian method.

This equivalence is interesting for a few reasons. In the next section we will discuss the so called split Bregman iteration. This variant of the Bregman algorithm leads to problems of the form

$$\arg\min_{u,d} \; J(u,d) \quad \text{such that} \quad d - Au - b = 0 \tag{3.84}$$

with a convex function J, a matrix A and a vector b. The split Bregman formulation then proceeds by solving this optimisation problem with the help of the Bregman iteration from the previous sections. Interestingly it is also the same kind of problem for which we have shown the equivalence between the Bregman framework and the augmented Lagrangian method. It follows, that these three methods are closely linked to each other and that they share a rather large set of common properties. Furthermore, the split Bregman algorithm will also be the basis of our optical flow algorithms presented in Chapter 4. As a consequence, all the forthcoming work could also be interpreted as an application of the Lagrangian penalty method.

3.2 The split Bregman algorithm

The Bregman formulations that we have seen in Section 3.1 and Section 3.1.3 were limited to constrained optimisation problems. Our next step will be to show that there exists an elegant extension of the Bregman framework that makes it possible to minimise unconstrained convex energy functionals. This extension was proposed by Goldstein and Osher in [43] in 2009 and is known under the name *split Bregman algorithm*. Our presentation of the algorithm will be almost identical to the description found within this reference. We will, however, also discuss a certain number of extensions and variations that the original authors did not consider. The split Bregman formulation is based upon the Bregman algorithms from the previous sections. In Remark 3.24, we will show that there is little difference whether one considers the alternative Bregman formulation or whether one uses the standard form. This result cannot be found in [43]. Furthermore,

in Section 3.2.1 we will discuss how Theorem 3.17 can be applied to the split Bregman algorithm. This result is also new. The split Bregman formulation is especially useful for solving the following two problems:

$$\arg\min_u \ \|\Phi(u)\|_1 + H(u) \tag{3.85}$$

$$\arg\min_u \ \|\Phi(u)\|_2 + H(u) \tag{3.86}$$

In this context Φ is an affine mapping, i.e. $\Phi(u) = \Lambda u + b$ for some matrix Λ and some vector b. H should be a convex function from $\mathbb{R}^n$ to $\mathbb{R}$. The difficulty in minimising such cost functions lies within the fact that $\|\cdot\|_1$ and $\|\cdot\|_2$ are not differentiable in 0.

The basic idea behind the split Bregman approach is to introduce an additional variable that will enable us to separate the non-differentiable terms from the differentiable ones. This separation will make it possible to consider each term individually and to devise specific strategies that allow an efficient and accurate minimisation. Since the reasoning will be almost identical for eq. (3.85) and for eq. (3.86), we will limit us on eq. (3.85) and simply mention where both approaches would differ.

As already mentioned, the split Bregman algorithm introduces a new variable and transforms the unconstrained problem into a constrained problem. This is done by rewriting eq. (3.85) in the following way:

$$\arg\min_{d,u} \ \|d\|_1 + H(u) \quad \text{such that} \quad d = \Phi(u) \tag{3.87}$$

Clearly the formulations from eq. (3.85) and eq. (3.87) are equivalent. If we know the minimiser of eq. (3.85), then we can easily obtain the minimiser of eq. (3.87) and vice versa. In order to make the subsequent reasoning simpler and to avoid a cumbersome notation, we introduce the following expressions:

$$\begin{aligned} \eta &:= (u, d)^T \\ J(\eta) &:= \|d\|_1 + H(u) \\ A(\eta) &:= d - \Lambda u \end{aligned} \tag{3.88}$$

Obviously J is again a convex function and A is a linear map. These properties follow immediately from the properties of H and Φ. Using these notations, eq. (3.87) can be rewritten as

$$\arg\min_\eta \ J(\eta) \quad \text{such that} \quad \frac{1}{2}\|A(\eta) - b\|_2^2 = 0 \tag{3.89}$$

Equation (3.89) looks exactly like the kind of problems we analysed in the chapters before and can for example be solved with the alternative Bregman algorithm from Proposition 3.19. We assume at this point that it is possible to choose η^0 such that 0 is a subgradient of J at η^0. This is always possible if H attains its minimum. Functions like $H(u) := \exp(u)$ are not allowed although they are convex. By applying the alternative

formulation of the Bregman algorithm we obtain the following simple iterative procedure.

$$\begin{aligned} b^0 &= b \\ \eta^{k+1} &= \arg\min_{\eta} \; J(\eta) + \frac{\mu}{2}\left\|A(\eta) - b^k\right\|_2^2 \\ b^{k+1} &= b^k + b - A\eta^{k+1} \end{aligned} \tag{3.90}$$

with $\mu > 0$ being a constant positive parameter. Now we reintroduce the definitions of J and A and obtain the following iterative strategy

$$\begin{aligned} \left(u^{k+1}, d^{k+1}\right) &= \arg\min_{u,d} \; \|d\|_1 + H(u) + \frac{\mu}{2}\left\|d - \Lambda u - b^k\right\|_2^2 \\ b^{k+1} &= b^k + b - d^{k+1} + \Lambda u^{k+1} \end{aligned} \tag{3.91}$$

At first glance it is not clear why this formulation might be beneficial since the cost function contains now even more terms than before. Further, we have to minimise with respect to two variables instead of just one. The trick is to minimise the cost function in eq. (3.91) by alternating between the minimisation with respect to u and the minimisation with respect to d. Since we are dealing with a convex problem the alternating minimisation is actually a descent strategy and thus, we will obtain a sequence $(u^{k,j}, d^{k,j})_j$ that reduces the value of the cost function in each step and will sooner or later come close to the global minimum. Although this method is far from being efficient, the individual component functions can be minimized most of the time extremely fast. This compensates for the slowness of the componentwise minimisation. All in all, we have to perform the following two steps:

$$\begin{aligned} \text{Step 1} : u^{k,j+1} &= \arg\min_{u} \; H(u) + \frac{\mu}{2}\left\|d^{k,j} - \Lambda u - b^k\right\|_2^2 \\ \text{Step 2} : d^{k,j+1} &= \arg\min_{d} \; \|d\|_1 + \frac{\mu}{2}\left\|d - \Lambda u^{k,j+1} - b^k\right\|_2^2 \end{aligned} \tag{3.92}$$

The optimization of step 1 depends largely on the exact nature of H. As a consequence we cannot make any further claims about it at this point. We just note that for the case where $H(u) = \|Cu - f\|_2^2$ with some matrix C and a vector f, the cost function becomes differentiable and the minimiser can be obtained by solving a linear system of equations with a positive semi-definite matrix. If either C or Λ has full rank, then the system matrix will even be positive definite. In this case there exist many highly efficient algorithms. Step 2 is a well known problem. We analysed it in Section 2.4. As shown in Proposition 2.51 the solution is given by

$$d_i^{k,j+1} = \text{shrink}\left(\left(\Lambda u^{k,j+1} + b^k\right)_i, \frac{1}{\mu}\right) \tag{3.93}$$

where the $d_i^{k,j+1}$ are the components of the vector $d^{k,j+1}$. If we had used eq. (3.86) instead of eq. (3.85), then we would have to use the generalised shrinkage operator and apply Proposition 2.54. In that case, the solution is given by

$$d^{k,j+1} = \text{gshrink}\left(\Lambda u^{k,j+1} + b^k, \frac{1}{\mu}\right) \tag{3.94}$$

This is the only significant difference between the approach for eq. (3.85) and for eq. (3.86). Since the shrinkage operators only require elementary operations they can be carried out extremely fast.

The detailed formulation of the split Bregman algorithm with N iterations and M alternating minimisation steps is depicted in Algorithm 3.2.

Data: Λ, b, H, N, M
Result: u^N and d^N minimising eq. (3.87)
Initialize: u^0 such that $0 \in \partial H\left(u^0\right)$, $d^0 = 0$, $b^0 = b$
for $k = 0$ **to** $N - 1$ **do**
  $u^{k,0} = u^k$
  $d^{k,0} = d^k$
  for $j = 0$ **to** $M - 1$ **do**
    $u^{k,j+1} = \arg\min_u \; H(u) + \frac{\mu}{2}\left\|d^{k,j} - \Lambda u - b^k\right\|_2^2$
    $d^{k,j+1} = \arg\min_d \; \|d\|_1 + \frac{\mu}{2}\left\|d - \Lambda u^{k,j+1} - b^k\right\|_2^2$
  end
  $u^{k+1} = u^{k,M}$
  $d^{k+1} = d^{k,M}$
  $b^{k+1} = b^k + b - d^{k+1} + \Lambda u^{k+1}$
end

Algorithm 3.2: Split Bregman algorithm for N iterations with M alternating minimisation steps based upon the alternative form of the Bregman iteration.

At this point one should note that the split Bregman algorithm is simply a way of reformulating an unconstrained convex optimization problem in such a way that we receive a constrained optimization problem in the form of eq. (3.87). This problem is then solved with the Bregman algorithms that we saw in the previous sections. The crucial point is that this constrained optimisation problem is of the form

$$\arg\min_{u,d} \; J(u,d) \quad \text{sucht that} \quad \left\|A\binom{u}{d} - b\right\|_2^2 = 0 \tag{3.95}$$

with a convex function J, a matrix A and a vector b. Thus, it fits exactly into the category of problems for which we have a complete convergence theory. Propositions 3.8, 3.11 and 3.13 tell us that the iterates (u^k, d^k) converge towards a solution of $A\binom{u}{d} = b$ and that each iterate that solves this system also minimises the energy functional J. As a consequence we always have convergence towards a solution of eq. (3.87) and thus also towards a solution of eq. (3.85) (resp. eq. (3.86)).

Remark 3.24
The authors of [43] only considered the split Bregman iteration in combination with the alternative formulation of the Bregman algorithm. However, it would also be possible to

use the standard Bregman iteration for the deduction of the split Bregman algorithm. Our functional J in the split Bregman algorithm was of the form $J(u,d) = \|d\| + H(u)$. The corresponding Bregman divergence would look as follows:

$$\begin{aligned} D_J^{p_y}\left(\binom{x_1}{x_2}, \binom{y_1}{y_2}\right) &= J(x_1,x_2) - J(y_1,y_2) - \langle p_y, \binom{x_1}{x_2} - \binom{y_1}{y_2}\rangle \\ &= J(x_1,x_2) - J(y_1,y_2) - \langle p_{y_1}, x_1 - y_1\rangle - \langle p_{y_2}, x_2 - y_2\rangle \end{aligned} \tag{3.96}$$

Therefore, the split Bregman algorithm would require us to solve the following problem at each iteration step

$$\begin{aligned} \left(u^{k+1}, d^{k+1}\right) &= \arg\min_{u,d} J(u,d) - \langle p_u, u\rangle - \langle p_d, d\rangle + \frac{\lambda}{2}\|d - \Lambda u - b\|_2^2 \\ &= \arg\min_{u,d} \|d\| + H(u) - \langle p_u, u\rangle - \langle p_d, d\rangle + \frac{\lambda}{2}\|d - \Lambda u - b\|_2^2 \end{aligned} \tag{3.97}$$

The corresponding alternating minimisation scheme is given by

$$\begin{aligned} &\arg\min_u \; H(u) - \langle p_u, u\rangle + \frac{\lambda}{2}\|d - \Lambda u - b\|_2^2 \\ &\arg\min_d \; \|d\| - \langle p_d, d\rangle + \frac{\lambda}{2}\|d - \Lambda u - b\|_2^2 \end{aligned} \tag{3.98}$$

The scalar products are differentiable functions and will not cause any additional difficulties in the minimisation of the first expression. The second expression, however, requires a modification of the shrinkage operator. If $\|d\| = \|d\|_1$, then the minimisation can be reduced to 1-D problems of the form

$$\arg\min_{d_i} |d_i| - p d_i + \frac{\lambda}{2}(d_i - f)^2 \tag{3.99}$$

d_i is a minimiser if and only if it fulfils the equation

$$\begin{aligned} 0 &= \partial(|\cdot|)(d) - p + \lambda(d_i - f) \\ &= \partial(|\cdot|)(d) + \lambda\left(d_i - f - \frac{1}{\lambda}p\right) \end{aligned} \tag{3.100}$$

With exactly the same arguments as in the proof of Proposition 2.51 it follows that d_i is given by

$$d_i = \begin{cases} f + \frac{1}{\lambda}(p-1) & \text{if } f + \frac{p}{\lambda} \in \left(\frac{1}{\lambda}, \infty\right) \\ 0 & \text{if } f + \frac{p}{\lambda} \in \left[-\frac{1}{\lambda}, \frac{1}{\lambda}\right] \\ f + \frac{1}{\lambda}(p+1) & \text{if } f + \frac{p}{\lambda} \in \left(-\infty, -\frac{1}{\lambda}\right) \end{cases} \tag{3.101}$$

which corresponds to $d_i = \text{shrink}\left(f + \frac{1}{\lambda}p, \frac{1}{\lambda}\right)$. If $\|d\| = \|d\|_2$, then d must fulfil the following equation

$$0 = \partial\left(\|\cdot\|_2\right)(d) - p_d + \lambda(d - \Lambda u - b) \tag{3.102}$$

$$= \partial\left(\|\cdot\|_2\right)(d) + \lambda\left(d - \underbrace{\left(\Lambda u + b + \frac{1}{\lambda} p_d\right)}_{:=\tilde{f}}\right)$$

This is exactly the same equation that we considered in the proof of Proposition 2.54. It follows that d is given by

$$d = \operatorname{gshrink}\left(\tilde{f}, \frac{1}{\lambda}\right) = \operatorname{gshrink}\left(\Lambda u + b + \frac{1}{\lambda} p_d, \frac{1}{\lambda}\right) \tag{3.103}$$

To conclude, we can state that the split Bregman algorithm can also be formulated with the standard Bregman iteration. The complete algorithm is given in Algorithm 3.3. In the following we will continue to use the split Bregman algorithm based on the alternative Bregman iteration. One of the reasons for this choice is given in Remark 3.26.

Data: Λ, b, H, N, M
Result: u^N and d^N minimising eq. (3.87)
Initialize: u^0, d^0 arbitrary, $p^0 = \left(p_u^0, p_d^0\right)^T \in \partial\left(\|d\| + H(u)\right)\left(u^0, d^0\right)$
for $k = 0$ **to** $N - 1$ **do**
  $u^{k,0} = u^k$
  $d^{k,0} = d^k$
  for $j = 0$ **to** $M - 1$ **do**
    $u^{k,j+1} = \arg\min_u \; H(u) - \langle p_u^k, u\rangle + \frac{\mu}{2}\left\|d^{k,j} - \Lambda u - b\right\|_2^2$
    $d^{k,j+1} = \arg\min_d \; \|d\| - \langle p_d^k, d\rangle + \frac{\mu}{2}\left\|d - \Lambda u^{k,j+1} - b\right\|_2^2$
  end
  $u^{k+1} = u^{k,M}$
  $d^{k+1} = d^{k,M}$
  $p_u^{k+1} = p_u^k + \mu\Lambda^T\left(d^{k+1} - \Lambda u^{k+1} - b\right)$
  $p_d^{k+1} = p_d^k - \mu\left(d^{k+1} - \Lambda u^{k+1} - b\right)$
end

Algorithm 3.3: Split Bregman algorithm for N iterations with M alternating minimisation steps based upon the standard Bregman iteration.

Remark 3.25
The split Bregman algorithm, as presented above, also applies to problems of the following form

$$\arg\min_u \; \sum_{k=1}^{m} \|\Phi_k(u)\|_1 + H(u) \tag{3.104}$$

$$\arg\min_u \; \sum_{k=1}^{m} \|\Phi_k(u)\|_2 + H(u) \tag{3.105}$$

with $m > 1$ and where the same assumptions hold for the Φ_k and H as for eq. (3.85) and eq. (3.86). Instead of one variable d, we will use m variables d_k this time. Then one can reformulate eq. (3.104) as

$$\underset{u,d_1,\ldots,d_m}{\arg\min} \sum_{k=1}^{m} \|d_k\|_1 + H(u) \text{ such that } \frac{1}{2}\sum_{k=1}^{m} \|d_k - \Phi_k(u)\|_2^2 = 0 \tag{3.106}$$

In order to get an expression of the form of eq. (3.89), we take the following approach: we define the vector $\boldsymbol{d} := (d_1, d_2, \ldots, d_m)^T$ by stacking all the variables d_k on top of each other. Since the Φ_k are all affine mappings, they can each be described by a matrix Λ_k and a vector b_k which we can also stack on top of each other to obtain a very large matrix $\boldsymbol{\Lambda} := (\Lambda_1, \ldots, \Lambda_m)^T$ and a vector $\boldsymbol{b} := (b_1, \ldots, b_m)^T$. Then it is a trivial computation to see that

$$\sum_{k=1}^{m} \|d_k - \Phi_k(u)\|_2^2 = \|\boldsymbol{d} - \boldsymbol{\Lambda}u - \boldsymbol{b}\|_2^2 \tag{3.107}$$

and thus, eq. (3.106) can be reformulated as a problem of the form of eq. (3.89). As a consequence, the alternative formulation for the Bregman iteration can also be applied to this problem and all the nice properties like convergence remain verified. It remains to analyse how the formulation of the algorithm is affected by the sum. By simply replacing every occurrence of $\|d\|_1$ by $\sum_{k=1}^{m} \|d_k\|_1$ in the presentation above, we see that there is practically no change in the algorithm. The main difference is that we will have to apply the shrinkage on every variable d_k separately and that we will have to handle several variables b_k. The corresponding formulation is presented in Algorithm 3.4. Obviously the same argumentation also holds for eq. (3.105). All one has to do, is to exchange the soft shrinkage operator with the generalised shrinkage operator. The case $m = 2$ of this formulation has been discussed in [43] in the context of denoising methods with the ROF model.

Remark 3.26
The split Bregman algorithm, as presented in Algorithm 3.2 and Algorithm 3.4, requires an initialisation for u^0 that might be difficult to determine. However, by closely analysing the algorithms, one notices u^0 is actually never used during the iterations. Thus, it is basically enough to prove that such a value exists and to set u^0 to 0. Note, that the existence is really necessary, otherwise the alternative Bregman formulation cannot be applied. Also note, that this would not hold anymore if we would first minimise with respect to d and only then with respect to u. The order in which the variables are minimised is important. This observation certainly presents an advantage over the formulation given in Algorithm 3.3, where we used the standard Bregman iteration. It is not possible to avoid the (potentially highly non-trivial) computation of p^0 in Algorithm 3.3.

Remark 3.27
In theory, the parameter μ that appears in the above Bregman algorithms can be chosen arbitrarily. We showed in Section 3.1.2 that we have convergence as long as μ is positive. However, this parameter μ appears always inside the shrinkage operators during the

Data: Λ_k, b_k, for $k = 1, \ldots, m$, H, N, M
Result: u^N and d_k^N minimising eq. (3.106)
Initialize: u^0 such that $0 \in \partial H\left(u^0\right)$, $d_k^0 = 0$, $b_k^0 = b_k$ for $k = 1, \ldots, m$
for $i = 0$ **to** $N - 1$ **do**
  $u^{i,0} = u^i$
  $d_k^{i,0} = d_k^i$ for $k = 1, \ldots, m$
  for $j = 0$ **to** $M - 1$ **do**

$$u^{i,j+1} = \arg\min_u \; H(u) + \frac{\mu}{2} \sum_{k=1}^{m} \left\| d_k^{i,j} - \Lambda_k u - b_k^i \right\|_2^2$$

$$d_k^{i,j+1} = \arg\min_d \; \|d\|_1 + \frac{\mu}{2} \left\| d - \Lambda_k u^{i,j+1} - b_k^i \right\|_2^2 \text{ for } k = 1, \ldots, m$$

  end
  $u^{i+1} = u^{i,M}$
  $d_k^{i+1} = d_k^{i,M}$ for $k = 1, \ldots, m$
  $b_k^{i+1} = b_k^i + b_k - d_k^{i+1} + \Lambda_k u^{i+1}$ for $k = 1, \ldots, m$
end

Algorithm 3.4: Variant of the split Bregman algorithm for N iterations with M alternating minimisation steps.

alternating minimisation scheme of the split Bregman formulation. Therefore, setting it too small would result in setting the variables d always to 0. This might lead to a serious reduction in the convergence speed or possibly cause other problems.

As for the two previous Bregman algorithms, we also present a simple example on how the split Bregman algorithm can be used.

Example 3.28
Assume A and B are matrices, b and c two vectors and $\lambda > 0$. We wish to solve the following problem

$$\arg\min_x \|Ax + b\|_2 + \frac{\lambda}{2} \|Bx - c\|_2^2 \tag{3.108}$$

The split Bregman algorithm transforms this problem into the constrained form

$$\arg\min_{x,d} \|d\|_2 + \frac{\lambda}{2} \|Bx - c\|_2^2 \quad \text{such that} \quad \frac{1}{2} \|d - Ax - b\|_2^2 = 0 \tag{3.109}$$

and yields the following iterative strategy

$$\left(x^{k+1}, d^{k+1}\right) = \arg\min_{x,d} \|d\|_2 + \frac{\lambda}{2} \|Bx - c\|_2^2 + \frac{\mu}{2} \left\| d - Ax - b^k \right\|_2^2 \tag{3.110}$$

$$b^{k+1} = b^k + b - d^{k+1} + Ax^{k+1}$$

with $x^0 = 0$, $d^0 = 0$ and $b^0 = b$. Note that this choice is possible because of Remark 3.26. $\frac{\lambda}{2} \|Bx - c\|_2^2$ attains its minimum if x is a solution of $B^T Bx = B^T c$. The next step consists in minimising the cost function alternatively with respect to x and with respect to d. The minimisation with respect to x comes down to solving the following linear system

$$\left(\lambda B^T B + \mu A^T A\right) x = \lambda B^T c + \mu A^T \left(d^k - b^k\right) \tag{3.111}$$

whereas for d it is enough to apply the generalised shrinkage operators. It follows that the complete iterative procedure looks as in Algorithm 3.5.

Data: A, B, b, c, N, M
Result: x^N minimising $\|Ax + b\|_2 + \frac{\lambda}{2} \|Bx - c\|_2^2$
Initialize: $x^0 = 0$, $d^0 = 0$, $b^0 = b$
for $i = 0$ **to** $N - 1$ **do**
  $x^{i,0} = x^i$
  $d^{i,0} = d^i$
  for $j = 0$ **to** $M - 1$ **do**
    Solve $\left(\lambda B^T B + \mu A^T A\right) x = \lambda B^T c + \mu A^T \left(d^{i,j} - b^i\right) \longrightarrow x^{i,j+1}$
    $d^{i,j+1} = \text{gshrink}\left(Ax^{i,j+1} + b^i, \frac{1}{\mu}\right)$
  end
  $x^{i+1} = x^{i,M}$
  $d^{i+1} = d^{i,M}$
  $b^{i+1} = b^i + b - d^{i+1} + Ax^{i+1}$
end

Algorithm 3.5: Example of the split Bregman algorithm.

3.2.1 Convergence speed of the split Bregman algorithm

Since the split Bregman algorithm, as we consider it in this thesis, is based upon the alternative formulation of the Bregman algorithm, we know that all the results about the convergence that we have seen about that algorithm also hold here. In particular, Theorem 3.17 must hold. This theorem gave us an estimate for the convergence speed if certain regularity conditions were met. In the following, we would like to analyse if these conditions can be fulfilled for the split Bregman algorithm. We are going to consider the following problem

$$\arg\min_u \sum_{k=1}^{N} \|A_k u + b_k\|_2 + \frac{\lambda}{2} \|Bu - c\|_2^2 \tag{3.112}$$

where $A_k : \mathbb{R}^n \to \mathbb{R}^m$, $B : \mathbb{R}^n \to \mathbb{R}^l$ are matrices, $b_k \in \mathbb{R}^m$, $c \in \mathbb{R}^l$ some vectors and $\lambda > 0$ a real-valued parameter. This problem corresponds to those mentioned in Remark 3.25

with $\Phi_k(u) = A_k u + b_k$ and $H(u) = \frac{\lambda}{2}\|Bu - c\|_2^2$. After introducing the additional variables, eq. (3.112) becomes

$$\underset{u,d_1,\ldots,d_N}{\arg\min} \sum_{k=1}^{N} \|d_k\|_2 + \frac{\lambda}{2}\|Bu - c\|_2^2 \quad \text{such that} \quad \sum_{k=1}^{N} \|d_k - A_k u - b_k\|_2^2 = 0 \tag{3.113}$$

Note, that the necessary conditions for the application of the split Bregman algorithm are met. The cost function attains its global minimum for $d_k = 0$ for all k and u a solution of $B^T Bu = B^T c$. The constraining condition obviously also has a solution. Let us now define the matrix $\Lambda : \mathbb{R}^{n+Nm} \to \mathbb{R}^{Nm}$ by

$$\Lambda := \begin{pmatrix} -A_1 & I & 0 & \ldots & 0 \\ -A_2 & 0 & I & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -A_N & 0 & 0 & \ldots & I \end{pmatrix} \tag{3.114}$$

where I is the identity matrix in $\mathbb{R}^m$. If we define further $\tilde{b} = (b_1, b_2, \ldots, b_N)^T$, then eq. (3.113) can be rewritten as

$$\underset{u,d_1,\ldots,d_N}{\arg\min} \underbrace{\sum_{k=1}^{N} \|d_k\|_2 + \frac{\lambda}{2}\|Bu - c\|_2^2}_{:=J(u,d_1,\ldots,d_N)} \quad \text{such that} \quad \left\|\Lambda (u, d_1, \ldots, d_N)^T - \tilde{b}\right\|_2^2 = 0 \tag{3.115}$$

Now assume, that we have found $\boldsymbol{u} = (\tilde{u}, \tilde{d_1}, \ldots, \tilde{d_N})^T$, a J minimising solution of $\Lambda x = \tilde{b}$. In order to apply Theorem 3.17, we need to know how $\partial J(\tilde{u}, \tilde{d_1}, \ldots, \tilde{d_N})$ looks like. So assume, that $(w, w_1, \ldots, w_N)^T$ is a subgradient. By definition, we must have for all $d_k,\ k = 1, \ldots, N$ and all u:

$$\begin{aligned} \sum_{k=1}^{N} \|d_k\|_2 + \frac{\lambda}{2}\|Bu - c\|_2^2 - \sum_{k=1}^{N} \left\|\tilde{d_k}\right\|_2 + \frac{\lambda}{2}\|B\tilde{u} - c\|_2^2 \\ \geqslant \langle w, u - \tilde{u}\rangle + \sum_{k=1}^{N} \langle w_k, d_k - \tilde{d_k}\rangle \end{aligned} \tag{3.116}$$

Since this must hold for all possible choices, it must hold especially for $d_k = \tilde{d_k}$ with $k = 1, \ldots, N$. But then we see, that w must be a subgradient of $\frac{\lambda}{2}\|Bu - c\|_2^2$ at $\tilde{u}$. Setting $u = \tilde{u}$ and all but one d_k to $\tilde{d_k}$ yields in the same way that every w_k must be a subgradient of $\|d\|_2$ at $\tilde{d_k}$. From Corollary 2.49 it follows, that we have the following representation

$$w = \lambda B^T (B\tilde{u} - c) \tag{3.117}$$

$$w_k = \frac{\tilde{d_k}}{\left\|\tilde{d_k}\right\|_2} \qquad \text{for } k = 1, \ldots, N \tag{3.118}$$

We assume here, that all $\tilde{d}_k$ are different from 0. If this is not the case, then the choice of the subgradient is not unique anymore and would complicate the following discussion. Theorem 3.17 requires that there is a vector $\boldsymbol{\gamma} := (\gamma_1, \ldots, \gamma_N)^T$ such that $\Lambda^T \boldsymbol{\gamma} \in \partial J(\tilde{u}, \tilde{d}_1, \ldots, \tilde{d_N})$. The matrix Λ^T looks as follows

$$\Lambda^T := \begin{pmatrix} -A_1^T & -A_2^T & \ldots & -A_N^T \\ I & 0 & \ldots & 0 \\ 0 & I & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & I \end{pmatrix} \tag{3.119}$$

and therefore the following conditions must be fulfilled

$$\Lambda^T \boldsymbol{\gamma} = \begin{pmatrix} -\sum_{k=1}^N A_k^T \gamma_k \\ \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_N \end{pmatrix} = \begin{pmatrix} \lambda B^T (B\tilde{u} - c) \\ \frac{\tilde{d}_1}{\left\|\tilde{d}_1\right\|_2} \\ \frac{\tilde{d}_2}{\left\|\tilde{d}_2\right\|_2} \\ \vdots \\ \frac{\tilde{d_N}}{\left\|\tilde{d_N}\right\|_2} \end{pmatrix} \tag{3.120}$$

But this equation is equivalent to

$$\sum_{k=1}^N A_k^T \frac{\tilde{d}_k}{\left\|\tilde{d}_k\right\|_2} = \lambda B^T (c - B\tilde{u}) \tag{3.121}$$

If this relation holds for the minimising solution, then the estimate given in Theorem 3.17 also holds for the split Bregman algorithm.

Remark 3.29
Should any of the $\tilde{d}_k$ be 0, then Corollary 2.49 states that any vector with norm less or equal than 1 would be a valid subgradient of $\|d\|_2$ at $\tilde{d}_k$. In that case, we gain additional degrees of freedom in the above formula and increase the chances that it can be fulfilled. Further, if eq. (3.112) has a minimiser $\tilde{u}$ such that the cost function becomes 0, then it follows from eq. (3.113), that all the $\tilde{d}_k$ can be set to 0. Therefore, it might be appropriate to adopt the convention $\frac{0}{0} = 0$. In this setting, eq. (3.121) could be reduced to the trivial equation $0 = 0$.

Remark 3.30
Equation (3.121) is basically of no use for practical purposes as it requires the knowledge of the exact solution in order to give an estimate for the convergence speed. Also note that the split Bregman algorithm still converges, even if eq. (3.121) is not fulfilled. Theorem 3.17 only gives an estimate for the convergence speed, not for the convergence itself. The convergence is guaranteed by Proposition 3.8, Proposition 3.11 and Proposition 3.13 as mentioned in the beginning of Section 3.2.

3.3 Summary and concluding remarks

In this section we presented mainly two versions of the Bregman iterative algorithm. The classical formulation in Section 3.1 solves constrained problems of the form

$$\arg\min_u \; J(u) \quad \text{such that} \quad H(u) = 0 \tag{3.122}$$

by computing iteratively

$$u^{k+1} = \arg\min_u \; D_J^{p^k}\left(u, u^k\right) + \lambda H(u) \tag{3.123}$$

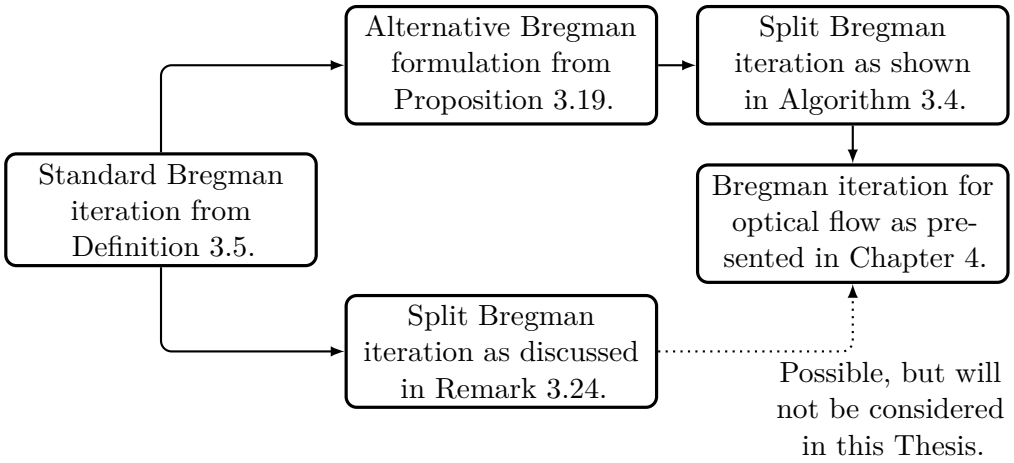
Furthermore, in Section 3.1 we discussed an alternative formulation, that does not require the computation of subgradients if H is of the form $\|Au - b\|_2^2$. This alternative formulation also plays a crucial role in the split Bregman algorithm which we presented in Section 3.2. The split Bregman algorithm solves variational problems of the form

$$\arg\min_u \; \|Au + b\| + H(u) \tag{3.124}$$

by applying the normal Bregman algorithm to the following equivalent constrained problem

$$\arg\min_{u,d} \; \|d\| + H(u) \quad \text{such that} \quad \|d - Au - b\| = 0 \tag{3.125}$$

The split Bregman algorithm is especially interesting as it allows us to minimise non-smooth convex energy functionals. Furthermore, we have presented a detailed convergence theory in Section 3.1.2 that applies to the standard Bregman iteration as well as the split Bregman approach. We were able to prove that both algorithms converge to the desired solutions and that it is possible to give estimates for the error if certain conditions are met. The convergence theory is especially important as it provides a mathematically sound basis for the optical flow problem that we wish to solve in the coming chapter. Finally we also proved an interesting link between the split Bregman algorithm and the augmented Lagrangian penalty method in Section 3.1.4. This equivalence allows us to interpret the forthcoming optical flow algorithms from different point of views. The following graphic summarises the development of the algorithms presented in this chapter. The arrows indicate which formulation can be deduced from which algorithm.



4 The Bregman iteration for optical flow

As we have seen in the previous chapters, the Bregman iteration possesses, at least from the theoretical point of view, several advantages over traditional penalty function/continuation methods and appears to be an interesting alternative when it comes to solving convex optimisation problems. Especially the split Bregman algorithm seems to be appealing because it allows us to minimise non-smooth cost functions. Our next goal will be to investigate how well the Bregman formulation really performs in practice. The main objective will be the accurate estimation of the displacement field for a given image sequence, also known as the optical flow problem.

In the following we will present a novel approach to solve the optical flow problem based upon the results from the previous chapters. Especially the split Bregman iteration will play a key-role role in our algorithms. The modelling part will be based mostly upon [18, 19, 20, 35, 78, 86, 95]. Further references will be cited when necessary.

Hereinafter we will denote our image sequence by a function $f : \Omega \times T \to \mathbb{R}$, where Ω is a subset of $\mathbb{R}^2$ representing the image domain (which in most cases can be assumed to be rectangular) and $T \subseteq \mathbb{R}$ should denote the temporal dimension. We wish to determine the flow field of f between two consecutive frames at the moments t and $t + 1$. The two components of the vector displacement field will be denoted by $u, v : \Omega \to \mathbb{R}$. Since f maps to $\mathbb{R}$, it is clear that we restrict ourselves to grey value images. Extensions to color valued images are possible but they render the modeling and the formulation of the algorithms cumbersome.

4.1 Problem formulation

As already mentioned, the optical flow problem seeks the displacement vector field that yields the correspondences between the pixels of two images. Figure 4.1 visualises this idea. This figure depicts two frames of an image sequence containing one moving and one static object. The task consists in determining those vectors that indicate where a given pixel from the first frame can be found in the second frame. Although this problem sounds rather simple, it is usually highly non-trivial. Solutions may or may not always exist. Even if they exist, they are not necessarily unique. Beside the solution already depicted in Fig. 4.1, one could also have mapped the upper end of the bar in the first frame to the lower end in the second frame and subsequently the lower end of the bar in frame one to the upper end of the bar in frame two. This is a valid solution and corresponds to a rotation by half a turn followed by a translation. In general, the considered image sequences are based off real world data and are even more difficult to handle than this trivial example from Fig. 4.1. Large displacements, noise, occlusions and illumination changes are only a few of the problems that must be handled properly.

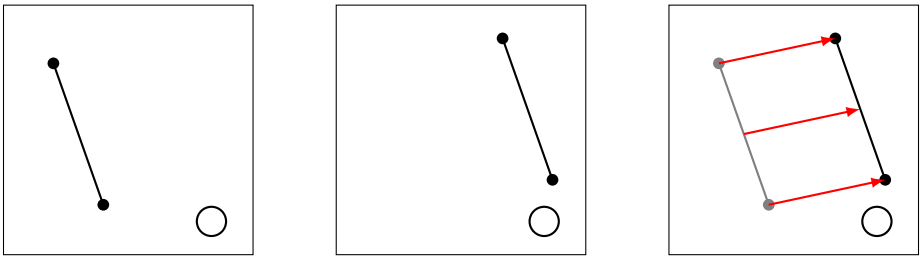


Figure 4.1: **Left:** Frame at time t. **Middle:** Frame at time $t+1$. **Right:** Overlaid frames with positions at time t marked in grey and sought displacement field in red.

As a consequence, it is important to chose a mathematical formulation that allows us to handle all these difficulties. A plethora of more or less suited formulations can be found in the literature. However, we will limit us to a single class, namely variational formulations.

Variational formulations allow a mathematical sound integration of different concepts into a single minimisation framework and therefore, they belong to the best performing and best understood techniques for solving the optical flow problem. They can easily be designed in such a way that they preserve motion boundaries, treat large displacements correctly, are robust with respect to illumination changes or perform favourably in the presence of noise and occlusions. See for example [5, 18, 19, 20, 35, 50, 95, 98]. Furthermore, they represent an almost optimal framework for the Bregman iteration. Variational methods are based on the minimisation of an energy functional. This energy functional is usually of the form

$$E(u,v) := \int_\Omega D\left(\partial^k f, u, v\right) + \lambda S\left(\nabla f, \nabla u, \nabla v\right) \mathrm{d}x \tag{4.1}$$

where $\lambda > 0$ is positive regularisation parameter and $\partial^k f$ the set of all partial derivatives of order k of the image sequence f. In this context D is often called the data term, whereas S is called smoothness term. In order to determine correspondences, we must assume that a certain quantity remains constant through time. This constancy assumption is usually stored inside the data term. Furthermore, it seems to be reasonable to assume that the displacement does not change significantly when we consider the neighboring pixels. In mathematical terms this means that we want the functions u and v to be smooth. This requirement is represented by the smoothness term. The parameter λ basically dictates how much we trust the different assumptions. Large values indicate that the displacement field should be smooth, whereas small values express our desire for fulfilling the constancy assumption as good as possible.

4.1.1 Modeling the data term

The data term expresses what part of the image remains constant through time. Several choices are possible.

Grey value constancy: The simplest assumption that one can make. Here one assumes that the following equality holds

$$f\left(x+u\left(x,y\right),y+v\left(x,y\right),t+1\right)=f\left(x,y,t\right) \tag{4.2}$$

Surprisingly, this assumption is relatively often fulfilled when the displacements remain small enough. Unfortunately, we have two unknowns but only one equation. Thus, there is no chance to recover the complete displacement field based on this equation alone. This problem is known in the literature as the aperture problem.

Gradient constancy: Assuming that the spatial gradient of f remains constant leads to the following equation

$$\nabla f\left(x+u\left(x,y\right),y+v\left(x,y\right),t+1\right)=\nabla f\left(x,y,t\right) \tag{4.3}$$

Here we have two unknowns and two equations. As a consequence, the aperture problem is not always present. Furthermore, the gradient constancy assumption remains fulfilled when the image undergoes global illumination changes, whereas the grey value constancy does not.

Other higher order assumptions: Alternative choices would be the constancy of the Hessian or the Laplacian of f. However, due to the high order of differentiation, these assumptions often suffer from stability problems, especially in presence of noise. As a consequence they are less attractive and rarely used.

Combinations of several constancy assumptions: It is possible to combine several constancy assumption into a single formulation. This way one obtains the strengths of each constancy assumption. The most popular choice is to enforce the constancy of the grey value and the constancy of the gradient. Although combining several assumptions easily leads to over determined systems, we will see that this does not cause any major problems.

The above models are all non-linear and will lead to non-convex energy functionals. However, as we have seen in the previous chapters, the Bregman formulation requires convex optimization problems. In order to overcome this hurdle, we approximate the left hand side of the equations above by their corresponding first order Taylor expansions. Then eq. (4.2) becomes

$$f_x\left(x,y,t\right)u\left(x,y\right)+f_y\left(x,y,t\right)v\left(x,y\right)+f_t\left(x,y,t\right)=0 \tag{4.4}$$

and eq. (4.3) becomes

$$f_{xx}\left(x,y,t\right)u\left(x,y\right)+f_{xy}\left(x,y,t\right)v\left(x,y\right)+f_{xt}\left(x,y,t\right)=0 \tag{4.5}$$

$$f_{xy}\left(x,y,t\right)u\left(x,y\right)+f_{yy}\left(x,y,t\right)v\left(x,y\right)+f_{yt}\left(x,y,t\right)=0 \tag{4.6}$$

where the indices designate the derivatives with respect to the corresponding variables. Note that this approximation is only accurate as long as the displacements are small.

In the following we will be using the combined linearised grey value and gradient constancy assumption for our purposes. This choice has proven in the past to be highly successful. See for example [18, 98]. The following two definitions for the data term are possible

$$D_1(u,v) = (f_x u + f_y v + f_t)^2 + \gamma\left((f_{xx}u + f_{xy}v + f_{xt})^2 + (f_{xy}u + f_{yy}v + f_{yt})^2\right) \quad (4.7)$$

$$D_2(u,v) = |f_x u + f_y v + f_t| + \gamma\Big(|f_{xx}u + f_{xy}v + f_{xt}| + |f_{xy}u + f_{yy}v + f_{yt}|\Big) \quad (4.8)$$

where the dependencies of the variables have been omitted for simplicity. γ is a positive weight between the two assumptions. It basically states which one is more important.

D_1 is interesting because it is convex and smooth. From a theoretical point of view it has optimal characteristics. However, it is not robust with respect to outliers. Already small violations of the constancy assumption could spoil the result due to the heavy penalisation. On the other side, D_2 is a much more robust term since the penalisation is sub-quadratic. However, its disadvantage is that the absolute value is not differentiable in 0.

4.1.2 Modeling the smoothness term

The smoothness term acts as a regularisation term. Inside homogeneous areas the data term suffers from the aperture problem (no matter which constancy assumption we consider as the image derivatives will always be 0, and thus every choice for u and v would be optimal). The smoothness term guarantees the uniqueness and existence of a solution even in such areas by extending the solution from non-homogeneous areas to the homogeneous ones and enforcing that the results remain smooth. This effect is also known as in-painting. Basically three smoothness terms are possible

$$S_1(u,v) := \|\nabla u\|_2^2 + \|\nabla v\|_2^2 \quad (4.9)$$

$$S_2(u,v) := \|\nabla u\|_2 + \|\nabla v\|_2 \quad (4.10)$$

$$S_3(u,v) := \sqrt{\|\nabla u\|_2^2 + \|\nabla v\|_2^2} \quad (4.11)$$

where we define

$$\|\nabla u\|_2 := \sqrt{(\partial_x u)^2 + (\partial_y u)^2} \quad (4.12)$$

The most interesting smoothness terms are S_2 and S_3. Both offer a sub-quadratic penalisation. Furthermore, S_3 is even rotationally invariant. Although S_1 is convex and differentiable and thus offers the best characteristics from a mathematical point of view, its weakness is the quadratic penalisation that causes a strong smoothing effect and that does not respect discontinuities in the motion field. The sub-quadratic penalisation of S_2 and S_3 on the other hand allows a conservation of these discontinuities. Therefore, we can expect that S_2 and S_3 will offer better results in terms of accuracy. However, they will be slightly more difficult to handle since they are not differentiable.

4.1.3 The optical flow models in the continuous setting

Now that we have presented the smoothness and data terms, we can combine them to different energy functionals. The following choices are the most interesting ones.

$$E_1(u,v) := \int_\Omega D_1(u,v) + \lambda S_1(u,v)\,\mathrm{d}x\mathrm{d}y \tag{4.13}$$

$$E_2(u,v) := \int_\Omega D_2(u,v) + \lambda S_1(u,v)\,\mathrm{d}x\mathrm{d}y \tag{4.14}$$

$$E_3(u,v) := \int_\Omega D_1(u,v) + \lambda S_2(u,v)\,\mathrm{d}x\mathrm{d}y \tag{4.15}$$

$$E_4(u,v) := \int_\Omega D_1(u,v) + \lambda S_3(u,v)\,\mathrm{d}x\mathrm{d}y \tag{4.16}$$

$$E_5(u,v) := \int_\Omega D_2(u,v) + \lambda S_2(u,v)\,\mathrm{d}x\mathrm{d}y \tag{4.17}$$

$$E_6(u,v) := \int_\Omega D_2(u,v) + \lambda S_3(u,v)\,\mathrm{d}x\mathrm{d}y \tag{4.18}$$

If we set $\gamma = 0$, then E_1 is also known as the model of Horn and Schunck. It dates back to 1981. See for example [50]. This model is extremely simple and can be solved very efficiently with the help of the corresponding Euler Lagrange equations. E_5 is sometimes also referred to as the TV-L_1 model. The authors of [66, 86, 95] analysed this model with a data term that only contained the grey value constancy and presented an efficient method to solve this optimisation problem. E_6 is a similar model that only differs in the fact that it is rotationally invariant. Energy functionals similar to E_3 have already been studied in 1993 by Isaac Cohen. Some of his results can be found in [35]. E_4 is a rotationally invariant formulation of the same model. The three energies E_2, E_3 and E_4 are exactly of the same type as the cost functions for which the split Bregman formulation was initially designed for. One term is differentiable whereas the other is not.

4.1.4 Discretizing the energy functionals

The Bregman algorithms, such as we have formulated them, only work in the discrete setting. Therefore, we must discretize the energy functionals from the previous section. For simplicity we will assume that our image has $N \times M$ pixels and that the image domain is given by $\Omega = [0,n] \times [0,m]$. The pixels should be distributed on an regular grid with distance h_x and h_y between two pixels in the respective direction marked by the index. As for the discretisation, there are countless methods to discretize an integral. The most simplest ones being the composite midpoint rule and the composite trapezoidal rule. Generally these formulas transform an integral into a finite sum of the following form

$$\int_\Omega g(x,y)\,\mathrm{d}x\mathrm{d}y \approx \sum_{i,j} c(i,j)\, g_{i,j} \tag{4.19}$$

where (i,j) denotes the coordinates at the points we have discretized the integral and $c(i,j)$ designates a positive weight. $g_{i,j}$ stands for $g(i,j)$. For the composite midpoint

rule, the $c(i,j)$ are all equal to $h_x h_y$, whereas for the trapezoidal rule they are given by

$$c(i,j) = \begin{cases} \frac{h_x h_y}{4} & \text{if } (i,j) \in \{1,N\} \times \{1,M\} \\ h_x h_y & \text{if } (i,j) \in \{2,\dots,N-1\} \times \{2,\dots,M-1\} \\ \frac{h_x h_y}{2} & \text{else} \end{cases} \tag{4.20}$$

A more detailed discussion of these formulas can also be found in [65, 76]. Using higher order formulas such as the composite Simpson rule would be possible but then we would have to restrict ourselves on images with an odd number of pixels in every direction. Since we do not want to abandon the freedom to have arbitrary numbers of pixels in our images we do not consider these approaches. Furthermore, as we can see from eq. (4.20), the weights for the trapezoidal rule and midpoint rule only differ along the boundaries. If the image contains a large number of pixels, the number of boundary pixels becomes insignificant. This suggests that the midpoint rule is a justified choice for discretizing the energy functionals. The loss in accuracy will probably be relatively small.

Discretizing the data terms

The different elements of the data term D_1 have the following discretisation:

$$\int_\Omega (f_x u + f_y v + f_t)^2 \, \mathrm{d}x\mathrm{d}y \approx h_x h_y \sum_{i,j} \left(f_{x\,i,j} u_{i,j} + f_{y\,i,j} v_{i,j} + f_{t\,i,j} \right)^2 \tag{4.21}$$

$$\int_\Omega (f_{xx} u + f_{xy} v + f_{xt})^2 \, \mathrm{d}x\mathrm{d}y \approx h_x h_y \sum_{i,j} \left(f_{xx\,i,j} u_{i,j} + f_{xy\,i,j} v_{i,j} + f_{xt\,i,j} \right)^2 \tag{4.22}$$

$$\int_\Omega (f_{yx} u + f_{yy} v + f_{yt})^2 \, \mathrm{d}x\mathrm{d}y \approx h_x h_y \sum_{i,j} \left(f_{yx\,i,j} u_{i,j} + f_{yy\,i,j} v_{i,j} + f_{yt\,i,j} \right)^2 \tag{4.23}$$

Whereas the elements of D_2 are given by

$$\int_\Omega |f_x u + f_y v + f_t| \, \mathrm{d}x\mathrm{d}y \approx h_x h_y \sum_{i,j} \left| f_{x\,i,j} u_{i,j} + f_{y\,i,j} v_{i,j} + f_{t\,i,j} \right| \tag{4.24}$$

$$\int_\Omega |f_{xx} u + f_{xy} v + f_{xt}| \, \mathrm{d}x\mathrm{d}y \approx h_x h_y \sum_{i,j} \left| f_{xx\,i,j} u_{i,j} + f_{xy\,i,j} v_{i,j} + f_{xt\,i,j} \right| \tag{4.25}$$

$$\int_\Omega |f_{yx} u + f_{yy} v + f_{yt}| \, \mathrm{d}x\mathrm{d}y \approx h_x h_y \sum_{i,j} \left| f_{yx\,i,j} u_{i,j} + f_{yy\,i,j} v_{i,j} + f_{yt\,i,j} \right| \tag{4.26}$$

If we regroup the components of the flow field in a single vector $(u,v)^T$ then eq. (4.21) can be written as $h_x h_y \left\| F \binom{u}{v} + \boldsymbol{f_t} \right\|_2^2$ with a matrix $F \in \mathbb{R}^{n_p \times 2n_p}$ given by

$$F := (D_x \mid D_y) \tag{4.27}$$

where D_x and D_y are two diagonal matrices that contain the pixels $f_{x\,i,j}$ (resp. $f_{y\,i,j}$) on their diagonal and n_p is the total number of pixels. $\boldsymbol{f_t}$ is the vector that contains the elements $f_{t\,i,j}$. Obviously one can define in the very same way matrices F_x and F_y such

that eq. (4.22) and eq. (4.23) can also be written as the squared 2-norm of a matrix vector product. Furthermore, the same idea can be applied to eqs. (4.24) to (4.26). Instead of using the squared 2-norm, one uses the 1-norm. Equation (4.24) would then become $h_x h_y \left\| F\binom{u}{v} + \boldsymbol{f_t} \right\|_1$.

Discretizing the smoothness terms

The discretisation of S_2 is straightforward. We obtain

$$\int_\Omega \|\nabla u\|_2 + \|\nabla v\|_2 \ \mathrm{d}x\mathrm{d}y \approx h_x h_y \sum_{i,j} \|\nabla u_{i,j}\|_2 + \|\nabla v_{i,j}\|_2 \tag{4.28}$$

Note that, in the discrete setting, the operator ∇ is simply a matrix, which we will designate for convenience by ∇ as well. This expression cannot be simplified any further. The same applies to S_3. We immediately obtain

$$\int_\Omega \sqrt{\|\nabla u\|_2^2 + \|\nabla v\|_2^2} \ \mathrm{d}x\mathrm{d}y \approx h_x h_y \sum_{i,j} \sqrt{\|\nabla u_{i,j}\|_2^2 + \|\nabla v_{i,j}\|_2^2} \tag{4.29}$$

For S_1 we obtain the following discretisation

$$\int_\Omega \|\nabla u\|_2^2 + \|\nabla v\|_2^2 \ \mathrm{d}x\mathrm{d}y \approx h_x h_y \sum_{i,j} (\partial_x u_{i,j})^2 + (\partial_y u_{i,j})^2 + (\partial_x v_{i,j})^2 + (\partial_y v_{i,j})^2 \tag{4.30}$$

$$= h_x h_y \left(\|\boldsymbol{\nabla}\, u\|_2^2 + \|\boldsymbol{\nabla}\, v\|_2^2 \right) \tag{4.31}$$

where $\boldsymbol{\nabla}$ is a matrix defined such that

$$\boldsymbol{\nabla}\, u := \begin{pmatrix} \partial_x u \\ \partial_y u \end{pmatrix} \tag{4.32}$$

$$\boldsymbol{\nabla}\, v := \begin{pmatrix} \partial_x v \\ \partial_y v \end{pmatrix} \tag{4.33}$$

holds. Note the difference between ∇ and $\boldsymbol{\nabla}$. ∇ operates on $u_{i,j}$ and returns the gradient of u at pixel (i, j). Thus $\nabla u_{i,j}$ is a vector in $\mathbb{R}^2$. On the other hand $\boldsymbol{\nabla}$ operates on u as a whole and returns a large vector containing $\partial_x u$ and $\partial_y u$ for every pixel. It follows that $\boldsymbol{\nabla}\, u \in \mathbb{R}^{2n_p}$, where n_p is again the number of pixels of our image. Both ∇ and $\boldsymbol{\nabla}$ are linear operators and thus can be represented as matrices. Unless otherwise stated, we will always approximate the first order derivatives of u and v with simple forward difference schemes.

4.1.5 The discrete optical flow models

Now that all the preparations are done, we can formulate the discrete optimisation problems that we wish to solve. To simplify the notation, we define

$$H_1(u, v) := \left\| F\binom{u}{v} + \boldsymbol{f_t} \right\|_2^2 + \gamma \left(\left\| F_x\binom{u}{v} + \boldsymbol{f_{xt}} \right\|_2^2 + \left\| F_y\binom{u}{v} + \boldsymbol{f_{yt}} \right\|_2^2 \right) \tag{4.34}$$

$$H_2(u,v) := \left\|F\binom{u}{v} + \boldsymbol{f_t}\right\|_1 + \gamma\left(\left\|F_x\binom{u}{v} + \boldsymbol{f_{xt}}\right\|_1 + \left\|F_y\binom{u}{v} + \boldsymbol{f_{yt}}\right\|_1\right) \tag{4.35}$$

where $\boldsymbol{f_{xt}}$ and $\boldsymbol{f_{yt}}$ are defined analogously to $\boldsymbol{f_t}$. Then our optical flow models from Section 4.1.3 correspond (in the same order) to the following convex optimisation problems

$$\arg\min_{u,v} H_1(u,v) + \frac{\lambda}{2}\left(\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2\right) \tag{4.36}$$

$$\arg\min_{u,v} H_2(u,v) + \frac{\lambda}{2}\left(\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2\right) \tag{4.37}$$

$$\arg\min_{u,v} \frac{\lambda}{2} H_1(u,v) + \sum_{i,j} \|\nabla u_{i,j}\|_2 + \|\nabla v_{i,j}\|_2 \tag{4.38}$$

$$\arg\min_{u,v} \frac{\lambda}{2} H_1(u,v) + \sum_{i,j} \sqrt{\|\nabla u_{i,j}\|_2^2 + \|\nabla v_{i,j}\|_2^2} \tag{4.39}$$

$$\arg\min_{u,v} \lambda H_2(u,v) + \sum_{i,j} \|\nabla u_{i,j}\|_2 + \|\nabla v_{i,j}\|_2 \tag{4.40}$$

$$\arg\min_{u,v} \lambda H_2(u,v) + \sum_{i,j} \sqrt{\|\nabla u_{i,j}\|_2^2 + \|\nabla v_{i,j}\|_2^2} \tag{4.41}$$

where $\frac{\lambda}{2}$ is the regularisation weight described in the beginning of this chapter. The different positions neither affect the model nor the algorithm. The regularisation parameter is placed in such a way that the formulas will be as elegant as possible later on. Furthermore, we are sometimes using $\frac{\lambda}{2}$ instead of λ. This will allow us to eliminate a certain number of multiplicative constants. Again, this choice has no influence on the algorithm. Finally, note that we were able to omit the factor $h_x h_y$ since we are only interested in the minimiser and not in the minimum itself.

4.2 Preprocessing steps

Before continuing with minimising our energy functional, we should have a look at the image sequence itself. In most cases it will be based off real world data. As a consequence, we must be able to deal with noise and other image degradations. Furthermore, there is no reason why f should be smooth or continuous, but such characteristics are generally of advantage, especially for theoretical purposes. Also, the first order Taylor approximation that we have performed in our approach requires the involved functions to be at least twice continuously differentiable. The remedy is to convolve f with a Gaussian kernel with a small standard deviation. This results in a smooth f and generally also eliminates a good portion of the noise. Furthermore, the Gaussian kernel acts as a low-pass filter which makes differentiation more stable.

4.3 Course of action for applying the Bregman algorithms

At this point we have presented all the energy functionals that we would like to consider. We will now demonstrate how the Bregman framework can be used to minimise these

formulations. The first model that we will discuss, will be the one of Horn and Schunck as formulated in eq. (4.36) and which corresponds to the energy E_1 from Section 4.1.3. It is our simplest approach and will allow us to conveniently introduce further necessary notations and to explain the general approach in a clearly understandable way. The respective Bregman formulation will be given in Section 4.4. Afterwards we will discuss the energy E_2, which is given in its discretized form by eq. (4.37) and which corresponds to a L_1-L_2 model. The Bregman algorithm belonging to this formulation will be given in Section 4.5. In Sections 4.6 and 4.7 we will analyse the L_2-L_1 models given by eqs. (4.38) and (4.39) and which correspond to the energies E_3 and E_4. As we will see, the Bregman approach for the isotropic formulation E_4 can easily be deduced from the algorithm for the anisotropic form E_3. Finally, in Section 4.8, we will discuss the two L_1-L_1 models represented by the energies E_5 and E_6 and which correspond to eqs. (4.40) and (4.41). In this case it will also be easy to deduce the isotropic Bregman formulation for eq. (4.41) from the anistropic approach for eq. (4.40). The minimisation of the L_1-L_1 functionals will require a small adaptation of the split Bregman algorithm. Although it consists of a rather simple trick, it cannot be found in the literature and therefore, it will also be presented in Section 4.8.

4.4 The model of Horn and Schunck

As already mentioned, the model of Horn and Schunck can be solved extremely fast if one combines the corresponding Euler-Lagrange equations with fast solvers for linear systems. In this section we will limit ourselves on showing how the split Bregman framework can be applied to the Horn and Schunck model. This means that we will set $\gamma = 0$ in the data term D_1 of the energy functional E_1. Already this simple case will make clear why the Bregman toolkit is not an optimal choice for this model. We will consider the problem

$$\arg\min_{u,v} \ \left\| F\binom{u}{v} + \boldsymbol{f_t} \right\|_2^2 + \frac{\lambda}{2} \left(\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2 \right) \tag{4.42}$$

with $\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2$ corresponding to the term H from eq. (3.105). The fact that the other term is the squared euclidean norm and not just the euclidean norm will be advantageous for our algorithm and not cause any difficulties. The first step of the split Bregman algorithm is to transform this expression into a constrained problem:

$$\arg\min_{u,v,d} \ \|d\|_2^2 + \frac{\lambda}{2} \left(\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2 \right) \text{ such that } \frac{1}{2} \left\| d - F\binom{u}{v} - \boldsymbol{f_t} \right\|_2^2 = 0 \tag{4.43}$$

Note that it does not matter which term is substituted. One could have substituted the smoothness term as well. Now we proceed exactly as for the split Bregman algorithm in Section 3.2. The next step will be to check whether there exist u^0, v^0, d^0 such that 0 is a subgradient of the cost-function. In this case, the verification of this requirement is trivial. It suffices to set all variables to 0. Then the cost function attains its minimum and 0 will be a subgradient. Furthermore, note that the constraining condition obviously also has a solution, namely $u = v = 0$ and $d = \boldsymbol{f_t}$. Thus all requirements are met and we

can almost immediately formulate the necessary equations as in eq. (3.91). The Bregman iteration asks us to find, at each iteration step, a solution of

$$\underset{u,v,d}{\arg\min}\ \|d\|_2^2 + \frac{\lambda}{2}\left(\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2\right) + \frac{\mu}{2}\left\|d - F\binom{u}{v} - \boldsymbol{f_t}\right\|_2^2 \tag{4.44}$$

We remind that it is not necessary to compute the Bregman divergence of the cost function from eq. (4.43) since we consider the split Bregman framework in combination with the alternative Bregman formulation as defined in Proposition 3.19. This observation will also will also be true for all the forthcoming algorithms.

The cost function in eq. (4.44) is convex and differentiable. Thus the minimum can be easily found through differentiation. We do not need an alternating minimisation scheme as the split Bregman algorithm suggests it. The differentiation with respect to d is simple and yields the equation

$$2d + \mu\left(d - F\binom{u}{v} - \boldsymbol{f_t}\right) = 0 \tag{4.45}$$

Differentiating the middle term of the sum is also simple. We observe that $\boldsymbol{\nabla}^T \boldsymbol{\nabla} = -\boldsymbol{\Delta}$, where $\boldsymbol{\Delta}$ is the discrete version of the Laplace operator. There are mainly two ways to see this. For the first one, one remembers that the adjoint operator of the gradient is the negative divergence. Then the conclusion follows from $\operatorname{div} \nabla = \Delta$. The second way is to carry out the matrix multiplication $\boldsymbol{\nabla}^T \boldsymbol{\nabla}$. Since we decided to use forward difference schemes, the matrix $\boldsymbol{\nabla}$ only contains two non-zero entries per row and a simple but lengthy computation shows that $\boldsymbol{\nabla}^T \boldsymbol{\nabla}$ corresponds to the Laplace operator where the second derivatives have been approximated by

$$\begin{aligned}(\partial_{xx} u)_{i,j} &\approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h_x^2} \\ (\partial_{yy} u)_{i,j} &\approx \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h_y^2}\end{aligned} \tag{4.46}$$

Differentiating the last term with respect to u and v gives us $\mu F^T\left(F\binom{u}{v} + \boldsymbol{f_t} - d\right)$. Thus it is necessary to know the structure of F^T and $F^T F$. Now, from eq. (4.27) we know that

$$F = (D_x \mid D_y) \tag{4.47}$$

and thus

$$F^T = \begin{pmatrix} D_x \\ D_y \end{pmatrix} \tag{4.48}$$

Therefore, it follows immediately that

$$F^T F = \begin{pmatrix} D_x D_x & D_x D_y \\ D_y D_x & D_y D_y \end{pmatrix} \tag{4.49}$$

Assembling all the pieces back together shows that we have to solve the following linear system

$$\mu\left(D_xD_xu + D_xD_yv + D_x\left(\boldsymbol{f_t} - d\right)\right) - \lambda\boldsymbol{\Delta}u = 0 \tag{4.50}$$
$$\mu\left(D_xD_yu + D_yD_yv + D_y\left(\boldsymbol{f_t} - d\right)\right) - \lambda\boldsymbol{\Delta}v = 0$$
$$2d + \mu\left(d - D_xu - D_yu - \boldsymbol{f_t}\right) = 0 \tag{4.51}$$

From the third equation it follows immediately, that

$$d = \frac{\mu}{2+\mu}\left(D_xu + D_yv + \boldsymbol{f_t}\right) \tag{4.52}$$

By plugging this into the first two equations, we obtain

$$\frac{2\mu}{2+\mu}\left(D_xD_xu + D_xD_yv + D_x\boldsymbol{f_t}\right) - \lambda\boldsymbol{\Delta}u = 0 \tag{4.53}$$
$$\frac{2\mu}{2+\mu}\left(D_xD_yu + D_yD_yv + D_y\boldsymbol{f_t}\right) - \lambda\boldsymbol{\Delta}v = 0$$

Finally, this means that the complete algorithm looks as in Algorithm 4.1

Data: D_x, D_y, $\boldsymbol{f_t}$, λ, μ, N
Result: u^N and v^N minimising eq. (4.42)
Initialize: $u^0 = v^0 = d^0 = 0$, ${\boldsymbol{f_t}}^0 = \boldsymbol{f_t}$
for $k = 0$ **to** $N - 1$ **do**
Solve:
$$\tfrac{2\mu}{2+\mu}\left(D_xD_xu + D_xD_yv + D_x{\boldsymbol{f_t}}^k\right) - \lambda\boldsymbol{\Delta}u = 0$$
$$\tfrac{2\mu}{2+\mu}\left(D_xD_yu + D_yD_yv + D_y{\boldsymbol{f_t}}^k\right) - \lambda\boldsymbol{\Delta}v = 0$$
Update:
$$d^{k+1} = \tfrac{\mu}{2+\mu}\left(D_xu^{k+1} + D_yv^{k+1} + {\boldsymbol{f_t}}^k\right)$$
$${\boldsymbol{f_t}}^{k+1} = {\boldsymbol{f_t}}^k + \boldsymbol{f_t} - d^{k+1} + D_xu^{k+1} + D_yv^{k+1}$$
end

Algorithm 4.1: Split Bregman algorithm for the Horn and Schunck model.

Remark 4.1
The system in eq. (4.53) is the very same linear system that one would have obtained by applying the Euler-Lagrange equations on the Horn and Schunck model. See for example [20, 57]. Thus it becomes immediately clear why the Bregman framework is not suited for energy functionals like this one. The Bregman algorithm requires us to solve a large linear system at every iteration step, whereas the Euler-Lagrange formulation only requires us to solve the same linear system a single time. On the other hand, it also shows that a single iteration with the Bregman iteration would already be enough to solve the problem. This might be a strong hint that the Bregman toolkit will be very efficient for more complicated models that contain non-differentiable terms.

4.5 The L_1-L_2 model

In this section we will show how the split Bregman algorithm can be used to solve

$$\arg\min_{u,v} \; H_2(u,v) + \frac{\lambda}{2}\left(\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2\right) \tag{4.54}$$

The notations are the same as in eq. (4.37). Our approach will be very similar to the strategy that we used for the Horn and Schunck model. In fact, $\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2$ will correspond again to the term named H in eq. (3.105). However, this time the data term contains non-differentiable parts. These will have to be transformed into a constraining condition for the split Bregman approach. This gives us immediately the following formulation

$$\begin{aligned} &\arg\min_{u,v,d,d_x,d_y} \; \|d\|_1 + \gamma\left(\|d_x\|_1 + \|d_y\|_1\right) + \frac{\lambda}{2}\left(\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2\right) \\ &\text{such that } \frac{1}{2}\left(\|d - F\tbinom{u}{v} - \boldsymbol{f_t}\|_2^2 + \|d_x - F_x\tbinom{u}{v} - \boldsymbol{f_{xt}}\|_2^2 + \|d_y - F_y\tbinom{u}{v} - \boldsymbol{f_{yt}}\|_2^2\right) = 0 \end{aligned} \tag{4.55}$$

Now, basically all that remains to be done is to verify whether the conditions to apply the alternative Bregman algorithm are met and to give a concrete formulation corresponding to Algorithm 3.4. The cost function reaches its minimum for $d = d_x = d_y = u = v = 0$, whereas the constraint can be fulfilled for $u = v = 0$ and $d = \boldsymbol{f_t}$, $d_x = \boldsymbol{f_{xt}}$ and $d_y = \boldsymbol{f_{yt}}$. Thus, it is possible to apply the split Bregman algorithm. The iterates are found through

$$\begin{aligned} &\arg\min_{u,v,d,d_x,d_y} \; \|d\|_1 + \gamma\left(\|d_x\|_1 + \|d_y\|_1\right) + \frac{\lambda}{2}\left(\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2\right) + \\ &\frac{\mu}{2}\left(\|d - F\tbinom{u}{v} - \boldsymbol{f_t}\|_2^2 + \|d_x - F_x\tbinom{u}{v} - \boldsymbol{f_{xt}}\|_2^2 + \|d_y - F_y\tbinom{u}{v} - \boldsymbol{f_{yt}}\|_2^2\right) \end{aligned} \tag{4.56}$$

by minimising alternatively with respect to u and v and with respect to d, d_x and d_y. Obviously the variables d, d_x and d_y can be updated efficiently with the soft shrinkage operator from Definition 2.50. Minimising with respect to u and v comes down to solving

$$\begin{aligned} \arg\min_{u,v} \frac{\mu}{2}&\left(\|d - F\tbinom{u}{v} - \boldsymbol{f_t}\|_2^2 + \|d_x - F_x\tbinom{u}{v} - \boldsymbol{f_{xt}}\|_2^2\right. \\ &\left. + \|d_y - F_y\tbinom{u}{v} - \boldsymbol{f_{yt}}\|_2^2\right) + \frac{\lambda}{2}\left(\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2\right) \end{aligned} \tag{4.57}$$

Differentiating leads almost immediately to the corresponding linear system that we must solve. Analogously to the matrices D_x and D_y that we used to define F, we now define diagonal matrices D_{xx} and D_{xy} for F_x and D_{yx} and D_{yy} for F_y that contain f_{xx}, f_{xy}, f_{yx} and f_{yy} respectively on their diagonals. With this notation and identical arguments as for the Horn and Schunck model, it follows that we have to solve the following linear system

$$\begin{aligned} &\left(D_x D_x + D_{xx} D_{xx} + D_{yx} D_{yx}\right) u + \left(D_x D_y + D_{xx} D_{xy} + D_{yx} D_{yy}\right) v \\ &\quad + D_x\left(\boldsymbol{f_t} - d\right) + D_{xx}\left(\boldsymbol{f_{xt}} - d_x\right) + D_{yx}\left(\boldsymbol{f_{yt}} - d_y\right) - \tfrac{\lambda}{\mu}\boldsymbol{\Delta} u = 0 \end{aligned} \tag{4.58}$$

$$\begin{aligned}(D_xD_y + D_{xx}D_{xy} + D_{yx}D_{yy})\,u + (D_yD_y + D_{xy}D_{xy} + D_{yy}D_{yy})\,v \\ + D_y\,(\boldsymbol{f_t} - d) + D_{xy}\,(\boldsymbol{f_{xt}} - d_x) + D_{yy}\,(\boldsymbol{f_{yt}} - d_y) - \tfrac{\lambda}{\mu}\boldsymbol{\Delta} v = 0\end{aligned} \tag{4.59}$$

This linear system is of similar structure as the corresponding system for the Horn and Schunck model. The complete algorithm for the L_1–L_2 model can now be formulated right away and is given in Algorithm 4.2. We observe that there is a high similarity

Data: F, F_x, F_y, $\boldsymbol{f_t}$, $\boldsymbol{f_{xt}}$, $\boldsymbol{f_{yt}}$, λ, γ, μ, N, M
Result: u^N and v^N minimising eq. (4.54)
Initialize: $u^0 = v^0 = d^0 = d_x^0 = d_y^0 = 0$, ${\boldsymbol{f_t}}^0 = \boldsymbol{f_t}$, ${\boldsymbol{f_{xt}}}^0 = \boldsymbol{f_{xt}}$, ${\boldsymbol{f_{yt}}}^0 = \boldsymbol{f_{yt}}$
for $k = 0$ **to** $N - 1$ **do**
$u^{k,0} = u^k$, $v^{k,0} = v^k$
$d^{k,0} = d^k$, $d_x^{k,0} = d_x^k$, $d_y^{k,0} = d_y^k$
for $j = 0$ **to** $M - 1$ **do**
Solve

$$\begin{aligned}\left(u^{k,j+1}, v^{k,j+1}\right) = \arg\min_{u,v} \frac{\mu}{2}\left\|d^{k,j} - F\tbinom{u}{v} - {\boldsymbol{f_t}}^k\right\|_2^2 + \frac{\mu}{2}\left\|d_x^{k,j} - F_x\tbinom{u}{v} - {\boldsymbol{f_{xt}}}^k\right\|_2^2 \\ + \frac{\mu}{2}\left\|d_y^{k,j} - F_y\tbinom{u}{v} - {\boldsymbol{f_{yt}}}^k\right\|_2^2 + \frac{\lambda}{2}\left(\|\boldsymbol{\nabla} u\|_2^2 + \|\boldsymbol{\nabla} v\|_2^2\right)\end{aligned}$$

by finding a solution of eq. (4.58) and eq. (4.59)

$$d^{k,j+1} = \operatorname{shrink}\left(D_x u^{k,j+1} + D_y v^{k,j+1} + {\boldsymbol{f_t}}^k, \frac{1}{\mu}\right) \text{ (done componentwise)}$$

$$d_x^{k,j+1} = \operatorname{shrink}\left(D_{xx} u^{k,j+1} + D_{xy} v^{k,j+1} + {\boldsymbol{f_{xt}}}^k, \frac{\gamma}{\mu}\right) \text{ (done componentwise)}$$

$$d_y^{k,j+1} = \operatorname{shrink}\left(D_{yx} u^{k,j+1} + D_{yy} v^{k,j+1} + {\boldsymbol{f_{yt}}}^k, \frac{\gamma}{\mu}\right) \text{ (done componentwise)}$$

end
$u^{k+1} = u^{k,M}$, $v^{k+1} = v^{k,M}$
$d^{k+1} = d^{k,M}$, $d_x^{k+1} = d_x^{k,M}$, $d_y^{k+1} = d_y^{k,M}$
${\boldsymbol{f_t}}^{k+1} = {\boldsymbol{f_t}}^k + \boldsymbol{f_t} - d^{k+1} + D_x u^{k+1} + D_y v^{k+1}$
${\boldsymbol{f_{xt}}}^{k+1} = {\boldsymbol{f_{xt}}}^k + \boldsymbol{f_{xt}} - d_x^{k+1} + D_{xx} u^{k+1} + D_{xy} v^{k+1}$
${\boldsymbol{f_{yt}}}^{k+1} = {\boldsymbol{f_{yt}}}^k + \boldsymbol{f_{yt}} - d_y^{k+1} + D_{yx} u^{k+1} + D_{yy} v^{k+1}$
end

Algorithm 4.2: The split Bregman algorithm for the L_1–L_2 model with N iterations and M alternating minimisation steps.

between solving this model and the previous Horn and Schunck model. As a matter of fact all the following algorithms will show this high similarity.

4.6 The non-rotationally invariant L_2-L_1 model

The next model that we will analyse has switched the differentiable and non-differentiable terms when compared to the model from Section 4.5. This time the data term is differentiable but the smoothness term is not. As we will see, this change does not present any problem for the Bregman framework. We remind that our goal is to minimise the following energy

$$\arg\min_{u,v} \ \frac{\lambda}{2} H_1(u,v) + \sum_{i,j} \|\nabla u_{i,j}\|_2 + \|\nabla v_{i,j}\|_2 \tag{4.60}$$

The notations are the same as in eq. (4.38). This case will also be an almost straightforward application of Algorithm 3.4. The only significant difference, when compared to the previous models, lies in the fact, that this time, the term H_1 from eq. (4.60) corresponds to H in eq. (3.105). As for the previous models, we transform the problem into a constrained formulation

$$\begin{aligned} \arg\min_{u,v} \ & \frac{\lambda}{2} H_1(u,v) + \sum_{i,j} \left\| d^u_{i,j} \right\|_2 + \left\| d^v_{i,j} \right\|_2 \\ \text{such that } & \frac{1}{2} \sum_{i,j} \left\| d^u_{i,j} - \nabla u_{i,j} \right\|_2^2 + \left\| d^v_{i,j} - \nabla v_{i,j} \right\|_2^2 = 0 \end{aligned} \tag{4.61}$$

and verify that the split Bregman algorithm can really be applied. The constraining condition admits a trivial solution and thus, it does not pose any problem. The cost function obviously has a minimum too. The variables $d^u_{i,j}$, $d^v_{i,j}$ act independently of u and v. Simply setting them all to 0 and determining the minimising u and v of H_1, by solving a least squares problem, yields the desired existence of a minimiser. This implies that the cost function attains its minimum and that there exists a point where 0 is a subgradient. Note that H_1 can always be minimised since we operate in a finite dimensional space, where such problems are always solvable. It follows that the split Bregman algorithm is applicable. We now make the following helpful observation

$$\|\boldsymbol{d^u} - \boldsymbol{\nabla}\, u\|_2^2 = \sum_{i,j} \left\| d^u_{i,j} - \nabla u_{i,j} \right\|_2^2 \tag{4.62}$$

where we regrouped all the vectors $d^u_{i,j}$ into one large vector $\boldsymbol{d^u}$. Note that the first half of $\boldsymbol{\nabla}\, u$ consists only of derivatives of u in x direction; thus, in order to have the right correspondences, the first half of $\boldsymbol{d^u}$ consists of the first components of all the vectors $d^u_{i,j}$ and the second half of all the second components of $d^u_{i,j}$. Therefore, we can rewrite eq. (4.61) in the following way

$$\begin{aligned} \arg\min_{u,v,\boldsymbol{d^u},\boldsymbol{d^v}} \ & \frac{\lambda}{2} H_1(u,v) + \sum_{i,j} \left\| d^u_{i,j} \right\|_2 + \left\| d^v_{i,j} \right\|_2 \\ & \text{such that } \frac{1}{2} \left(\|\boldsymbol{d^u} - \boldsymbol{\nabla}\, u\|_2^2 + \|\boldsymbol{d^v} - \boldsymbol{\nabla}\, v\|_2^2 \right) = 0 \end{aligned} \tag{4.63}$$

The corresponding Bregman iteration aims to solve

$$\begin{aligned}\underset{u,v,\boldsymbol{d^u},\boldsymbol{d^v}}{\arg\min}\ & \frac{\lambda}{2}\left(\left\|F\binom{u}{v}+\boldsymbol{f_t}\right\|_2^2+\gamma\left(\left\|F_x\binom{u}{v}+\boldsymbol{f_{xt}}\right\|_2^2+\left\|F_y\binom{u}{v}+\boldsymbol{f_{yt}}\right\|_2^2\right)\right)+ \\ & \sum_{i,j}\left\|d_{i,j}^u\right\|_2+\left\|d_{i,j}^v\right\|_2+\frac{\mu}{2}\left(\left\|\boldsymbol{d^u}-\boldsymbol{\nabla}\,u-\boldsymbol{b^u}\right\|_2^2+\left\|\boldsymbol{d^v}-\boldsymbol{\nabla}\,v-\boldsymbol{b^v}\right\|_2^2\right)\end{aligned} \tag{4.64}$$

Because of the alternating minimisation strategy, we will have to solve the following problem at each iteration step:

$$\begin{aligned}\underset{u,v}{\arg\min}\ & \frac{\lambda}{2}\left(\left\|F\binom{u}{v}+\boldsymbol{f_t}\right\|_2^2+\gamma\left(\left\|F_x\binom{u}{v}+\boldsymbol{f_{xt}}\right\|_2^2+\left\|F_y\binom{u}{v}+\boldsymbol{f_{yt}}\right\|_2^2\right)\right) \\ & +\frac{\mu}{2}\left(\left\|\boldsymbol{d^u}-\boldsymbol{\nabla}\,u-\boldsymbol{b^u}\right\|_2^2+\left\|\boldsymbol{d^v}-\boldsymbol{\nabla}\,v-\boldsymbol{b^v}\right\|_2^2\right)\end{aligned} \tag{4.65}$$

From what we have seen in the presentation of the last two models, it is immediately clear how the corresponding linear system must look like

$$\begin{aligned}&\left(D_xD_x+\gamma D_{xx}D_{xx}+\gamma D_{yx}D_{yx}\right)u+\left(D_xD_y+\gamma D_{xx}D_{xy}+\gamma D_{yx}D_{yy}\right)v \\ &\quad+D_x\boldsymbol{f_t}+\gamma D_{xx}\boldsymbol{f_{xt}}+\gamma D_{yx}\boldsymbol{f_{yt}}-\frac{\mu}{\lambda}\left(\boldsymbol{\Delta}u+\boldsymbol{\nabla}^T\left(\boldsymbol{d^u}-\boldsymbol{b^u}\right)\right)=0 \\ &\left(D_xD_y+\gamma D_{xx}D_{xy}+\gamma D_{yx}D_{yy}\right)u+\left(D_yD_y+\gamma D_{xy}D_{xy}+\gamma D_{yy}D_{yy}\right)v \\ &\quad+D_y\boldsymbol{f_t}+\gamma D_{xy}\boldsymbol{f_{xt}}+\gamma D_{yy}\boldsymbol{f_{yt}}-\frac{\mu}{\lambda}\left(\boldsymbol{\Delta}v+\boldsymbol{\nabla}^T\left(\boldsymbol{d^v}-\boldsymbol{b^v}\right)\right)=0\end{aligned} \tag{4.66}$$

Similarly as for the previous model, the variables $d_{i,j}^u$ and $d_{i,j}^v$ can be easily updated with the help of the generalised shrinkage operators from Definition 2.53. The complete algorithm is now given in Algorithm 4.3, where the indices k, l denote the coordinates of the pixels.

Remark 4.2
Observe that the model parameters λ and γ appear in the linear system and thus can influence directly the flow field components u and v. The L_1–L_2 model did not have that characteristic. There, the parameter γ could only influence the auxiliary variables directly. Its influence on u and v was indirect. Although this does not make a difference from the theoretical point of view, it might suggest that there will be a difference in how well the algorithms respond to parameter changes.

4.7 The rotationally invariant L_2-L_1 model

In this section we consider the following model:

$$\underset{u,v}{\arg\min}\ \frac{\lambda}{2}H_1\left(u,v\right)+\sum_{i,j}\sqrt{\left\|\nabla u_{i,j}\right\|_2^2+\left\|\nabla v_{i,j}\right\|_2^2} \tag{4.67}$$

The difference between this model and the model from Section 4.6 is that this one is rotationally invariant, whereas the other was not. Interestingly, the approach to minimise

Data: F, F_x, F_y, $\boldsymbol{f_t}$, $\boldsymbol{f_{xt}}$, $\boldsymbol{f_{yt}}$, λ, γ, μ, N, M
Result: u^N and v^N minimising eq. (4.60)
Initialize: $u^0 = v^0 = 0$, $d_{k,l}^{u,0} = 0$, $b_{k,l}^{u,0} = 0$, $d_{k,l}^{v,0} = 0$, $b_{k,l}^{v,0} = 0$ $\forall k, l$
for $i = 0$ **to** $N - 1$ **do**
$u^{i,0} = u^i$
$v^{i,0} = v^i$
$d_{k,l}^{u,i,0} = d_{k,l}^{u,i}$ for $\forall k, l$
$d_{k,l}^{v,i,0} = d_{k,l}^{v,i}$ for $\forall k, l$
for $j = 0$ **to** $M - 1$ **do**
Solve

$$\left(u^{i,j+1}, v^{i,j+1}\right) = \arg\min_{u,v} \frac{\lambda}{2} H_1(u, v) + \frac{\mu}{2}\left(\left\|\boldsymbol{d^{u,i,j}} - \boldsymbol{\nabla}\, u - \boldsymbol{b^{u,i}}\right\|_2^2 + \left\|\boldsymbol{d^{v,i,j}} - \boldsymbol{\nabla}\, v - \boldsymbol{b^{v,i}}\right\|_2^2\right)$$

by finding a solution of eq. (4.66)

$$d_{k,l}^{u,i,j+1} = \text{gshrink}\left(\nabla u_{k,l}^{i,j+1} + b_{k,l}^{u,i}, \frac{1}{\mu}\right) \ \forall k, l$$

$$d_{k,l}^{v,i,j+1} = \text{gshrink}\left(\nabla v_{k,l}^{i,j+1} + b_{k,l}^{v,i}, \frac{1}{\mu}\right) \ \forall k, l$$

end
$u^{i+1} = u^{i,M}$, $v^{i+1} = v^{i,M}$
$d_{k,l}^{u,i+1} = d_{k,l}^{u,i,M}$ for $\forall k, l$
$d_{k,l}^{v,i+1} = d_{k,l}^{v,i,M}$ for $\forall k, l$
$b_{k,l}^{u,i+1} = b_{k,l}^{u,i} - d_{k,l}^{u,i+1} + \nabla u_{k,l}^{i+1}$ for $\forall k, l$
$b_{k,l}^{v,i+1} = b_{k,l}^{v,i} - d_{k,l}^{v,i+1} + \nabla v_{k,l}^{i+1}$ for $\forall k, l$
end

Algorithm 4.3: The split Bregman algorithm for the non-rotationally invariant L_2–L_1 model.

both energies is almost identical. Before we start applying the Bregman algorithm, let us have a look at the smoothness term first. It can be reformulated in the following way

$$\begin{aligned}\sum_{i,j}\sqrt{\|\nabla u_{i,j}\|_2^2+\|\nabla v_{i,j}\|_2^2} &= \sum_{i,j}\sqrt{(\partial_x u_{i,j})^2+(\partial_y u_{i,j})^2+(\partial_x v_{i,j})^2+(\partial_y v_{i,j})^2} \\ &= \sum_{i,j}\left\|(\partial_x u_{i,j};\partial_y u_{i,j};\partial_x v_{i,j};\partial_y v_{i,j})^T\right\|_2 \\ &=: \sum_{i,j}\left\|\begin{pmatrix}\nabla u_{i,j}\\ \nabla v_{i,j}\end{pmatrix}\right\|_2\end{aligned} \tag{4.68}$$

Thus, this model can also be written in the following more compact form

$$\arg\min_{u,v}\ \frac{\lambda}{2}H_1(u,v)+\sum_{i,j}\left\|\begin{pmatrix}\nabla u_{i,j}\\ \nabla v_{i,j}\end{pmatrix}\right\|_2 \tag{4.69}$$

and is almost of the same form as the model from Section 4.6. The main difference is that, this time, we have one vector with four components for the smoothness term instead of two vectors with two components. The corresponding constraint formulation to eq. (4.67) is now easily deduced:

$$\arg\min_{u,v}\ \frac{\lambda}{2}H_1(u,v)+\sum_{i,j}\left\|\begin{pmatrix}d^u_{i,j}\\ d^v_{i,j}\end{pmatrix}\right\|_2 \text{ such that } \frac{1}{2}\sum_{i,j}\left\|\begin{pmatrix}d^u_{i,j}\\ d^v_{i,j}\end{pmatrix}-\begin{pmatrix}\nabla u_{i,j}\\ \nabla v_{i,j}\end{pmatrix}\right\|_2^2=0 \tag{4.70}$$

With the same reasoning as for the non-rotationally invariant L_2–L_1 model, it becomes obvious that the split Bregman algorithm is also applicable here. Now, notice the following detail

$$\begin{aligned}\sum_{i,j}\left\|\begin{pmatrix}d^u_{i,j}\\ d^v_{i,j}\end{pmatrix}-\begin{pmatrix}\nabla u_{i,j}\\ \nabla v_{i,j}\end{pmatrix}\right\|_2^2 &= \sum_{i,j}\left\|d^u_{i,j}-\nabla u_{i,j}\right\|_2^2+\left\|d^v_{i,j}-\nabla v_{i,j}\right\|_2^2 \\ &= \|\boldsymbol{d^u}-\boldsymbol{\nabla}\,u\|_2^2+\|\boldsymbol{d^v}-\boldsymbol{\nabla}\,v\|_2^2\end{aligned} \tag{4.71}$$

with the same definitions of $\boldsymbol{d^u}$ and $\boldsymbol{\nabla}\,u$ as before. The Bregman iteration can now be easily expressed by

$$\arg\min_{u,v,\boldsymbol{d^v},\boldsymbol{d^v}}\ \frac{\lambda}{2}H_1(u,v)+\sum_{i,j}\left\|\begin{pmatrix}d^u_{i,j}\\ d^v_{i,j}\end{pmatrix}\right\|_2+\frac{\mu}{2}\left(\|\boldsymbol{d^u}-\boldsymbol{\nabla}\,u-\boldsymbol{b^u}\|_2^2+\|\boldsymbol{d^v}-\boldsymbol{\nabla}\,v-\boldsymbol{b^v}\|_2^2\right) \tag{4.72}$$

Therefore, it follows that the necessary linear system we need to solve during the alternating minimisation scheme is exactly the same as in Section 4.6. The only difference between the two formulations lies in the minimisation with respect to the auxiliary variables. They can still be updated by using the generalised shrinkage operators, however, note that this time the vectors have four components each and not two as for the previous model. This allows us to give immediately the complete formulation as shown in Algorithm 4.4.

Data: F, F_x, F_y, $\boldsymbol{f_t}$, $\boldsymbol{f_{xt}}$, $\boldsymbol{f_{yt}}$, λ, γ, μ, N, M
Result: u^N and v^N minimising eq. (4.67)
Initialize: $u^0 = v^0 = 0$, $d_{k,l}^{u,0} = 0$, $b_{k,l}^{u,0} = 0$, $d_{k,l}^{v,0} = 0$, $b_{k,l}^{v,0} = 0$ $\forall k, l$
for $i = 0$ **to** $N - 1$ **do**
$u^{i,0} = u^i$
$v^{i,0} = v^i$
$d_{k,l}^{u,i,0} = d_{k,l}^{u,i}$ for $\forall k, l$
$d_{k,l}^{v,i,0} = d_{k,l}^{v,i}$ for $\forall k, l$
for $j = 0$ **to** $M - 1$ **do**
Solve

$$\left(u^{i,j+1}, v^{i,j+1}\right) = \arg\min_{u,v} \frac{\lambda}{2} H_1(u, v) + \frac{\mu}{2}\left(\left\|\boldsymbol{d^{u,i,j}} - \boldsymbol{\nabla}\, u - \boldsymbol{b^{u,i}}\right\|_2^2 + \left\|\boldsymbol{d^{v,i,j}} - \boldsymbol{\nabla}\, v - \boldsymbol{b^{v,i}}\right\|_2^2\right)$$

by finding a solution of eq. (4.66)

$$\begin{pmatrix} d_{k,l}^{u,i,j+1} \\ d_{k,l}^{v,i,j+1} \end{pmatrix} = \text{gshrink}\left(\begin{pmatrix} \nabla u_{k,l}^{i,j+1} \\ \nabla v_{k,l}^{i,j+1} \end{pmatrix} + \begin{pmatrix} b_{k,l}^{u,i} \\ b_{k,l}^{v,i} \end{pmatrix}, \frac{1}{\mu}\right) \;\forall\; k, l$$

end
$u^{i+1} = u^{i,M}$, $v^{i+1} = v^{i,M}$
$d_{k,l}^{u,i+1} = d_{k,l}^{u,i,M}$ for $\forall k, l$
$d_{k,l}^{v,i+1} = d_{k,l}^{v,i,M}$ for $\forall k, l$
$b_{k,l}^{u,i+1} = b_{k,l}^{u,i} - d_{k,l}^{u,i+1} + \nabla u_{k,l}^{i+1}$ for $\forall k, l$
$b_{k,l}^{v,i+1} = b_{k,l}^{v,i} - d_{k,l}^{v,i+1} + \nabla v_{k,l}^{i+1}$ for $\forall k, l$
end

Algorithm 4.4: The split Bregman algorithm for the rotationally invariant L_2–L_1 model.

4.8 The non-rotationally invariant L_1-L_1 model

The final model that we will discuss here is

$$\underset{u,v}{\arg\min}\ \lambda H_2(u,v) + \sum_{i,j} \|\nabla u_{i,j}\|_2 + \|\nabla v_{i,j}\|_2 \tag{4.73}$$

In contrast to the previous formulations, none of the terms of the energy functional is differentiable. This is unfortunate, because we need some differentiable parts that can coincide with H from eq. (3.105). The solution is to add a 0 to the cost function and to interpret it as a function $H(u) \equiv 0$. By using this little trick, it becomes possible again to apply Algorithm 3.4 and to proceed as before. First, we move all the non-differentiable terms into the constraining conditions. This leads us to

$$\begin{aligned}
&\underset{u,v,d,d_x,d_y,d^u_{i,j},d^v_{i,j}}{\arg\min}\ \lambda\Big(\|d\|_1 + \gamma\left(\|d_x\|_1 + \|d_y\|_1\right)\Big) + \sum_{i,j} \left\|d^u_{i,j}\right\|_2 + \left\|d^v_{i,j}\right\|_2 \qquad (4.74)\\
&\text{such that } \frac{1}{2}\left(\|d - F\tbinom{u}{v} - \boldsymbol{f_t}\|_2^2 + \|d_x - F_x\tbinom{u}{v} - \boldsymbol{f_{xt}}\|_2^2 + \|d_y - F_y\tbinom{u}{v} - \boldsymbol{f_{yt}}\|_2^2\right)\\
&\qquad + \frac{1}{2}\left(\sum_{i,j}\left\|d^u_{i,j} - \nabla u_{i,j}\right\|_2^2 + \left\|d^v_{i,j} - \nabla v_{i,j}\right\|_2^2\right) = 0
\end{aligned}$$

Or in the slightly more compact form

$$\begin{aligned}
&\underset{u,v,d,d_x,d_y,d^u_{i,j},d^v_{i,j}}{\arg\min}\ \lambda\Big(\|d\|_1 + \gamma\left(\|d_x\|_1 + \|d_y\|_1\right)\Big) + \sum_{i,j} \left\|d^u_{i,j}\right\|_2 + \left\|d^v_{i,j}\right\|_2 \qquad (4.75)\\
&\text{such that } \frac{1}{2}\left(\|d - F\tbinom{u}{v} - \boldsymbol{f_t}\|_2^2 + \|d_x - F_x\tbinom{u}{v} - \boldsymbol{f_{xt}}\|_2^2 + \|d_y - F_y\tbinom{u}{v} - \boldsymbol{f_{yt}}\|_2^2\right)\\
&\qquad + \frac{1}{2}\left(\|\boldsymbol{d^u} - \boldsymbol{\nabla}\, u\|_2^2 + \|\boldsymbol{d^v} - \boldsymbol{\nabla}\, v\|_2^2\right) = 0
\end{aligned}$$

Now we have to check whether the Bregman algorithm can be applied. The cost function has a trivial minimiser. Namely setting all the variables to 0. Furthermore, the constraining condition can also easily be solved by setting u and v to 0 and the auxiliary variables such that they cancel the remaining terms. Thus, we are in the setting to apply the split Bregman framework. It follows that we have to solve

$$\begin{aligned}
&\underset{u,v,d,d_x,d_y,d^u_{i,j},d^v_{i,j}}{\arg\min}\ \lambda\Big(\|d\|_1 + \gamma\left(\|d_x\|_1 + \|d_y\|_1\right)\Big) + \sum_{i,j} \left\|d^u_{i,j}\right\|_2 + \left\|d^v_{i,j}\right\|_2 \qquad (4.76)\\
&+\ \frac{\mu}{2}\left(\|d - F\tbinom{u}{v} - \boldsymbol{f_t}\|_2^2 + \|d_x - F_x\tbinom{u}{v} - \boldsymbol{f_{xt}}\|_2^2 + \|d_y - F_y\tbinom{u}{v} - \boldsymbol{f_{yt}}\|_2^2\right)\\
&\qquad + \frac{\mu}{2}\left(\|\boldsymbol{d^u} - \boldsymbol{\nabla}\, u - \boldsymbol{b^u}\|_2^2 + \|\boldsymbol{d^v} - \boldsymbol{\nabla}\, v - \boldsymbol{b^v}\|_2^2\right)
\end{aligned}$$

by minimising alternatively with respect to u and v and with respect to all the d-variables. As for all the other models, the minimisation with respect to u and v leads to a problem of the following form

$$\begin{aligned}
&\underset{u,v}{\arg\min}\ \|d - F\tbinom{u}{v} - \boldsymbol{f_t}\|_2^2 + \|d_x - F_x\tbinom{u}{v} - \boldsymbol{f_{xt}}\|_2^2 + \|d_y - F_y\tbinom{u}{v} - \boldsymbol{f_{yt}}\|_2^2 \qquad (4.77)\\
&\qquad + \|\boldsymbol{d^u} - \boldsymbol{\nabla}\, u - \boldsymbol{b^u}\|_2^2 + \|\boldsymbol{d^v} - \boldsymbol{\nabla}\, v - \boldsymbol{b^v}\|_2^2
\end{aligned}$$

All these terms already appeared in the previous models, such that it is easy now to deduce the corresponding linear system that gives us the minimiser of this expression. It is given by

$$\begin{aligned}
&\left(D_x D_x + D_{xx} D_{xx} + D_{yx} D_{yx}\right) u + \left(D_x D_y + D_{xx} D_{xy} + D_{yx} D_{yy}\right) v \\
&\quad + D_x \left(\boldsymbol{f_t} - d\right) + D_{xx} \left(\boldsymbol{f_{xt}} - d_x\right) + D_{yx} \left(\boldsymbol{f_{yt}} - d_y\right) - \left(\boldsymbol{\Delta} u + \boldsymbol{\nabla}^T \left(\boldsymbol{d^u} - \boldsymbol{b^u}\right)\right) = 0 \\
&\left(D_x D_y + D_{xx} D_{xy} + D_{yx} D_{yy}\right) u + \left(D_y D_y + D_{xy} D_{xy} + D_{yy} D_{yy}\right) v \\
&\quad + D_y \left(\boldsymbol{f_t} - d\right) + D_{xy} \left(\boldsymbol{f_{xt}} - d_x\right) + D_{yy} \left(\boldsymbol{f_{yt}} - d_y\right) - \left(\boldsymbol{\Delta} v + \boldsymbol{\nabla}^T \left(\boldsymbol{d^v} - \boldsymbol{b^v}\right)\right) = 0
\end{aligned} \tag{4.78}$$

The minimisation with respect to the d-variables is done once again with the help of the shrinkage operators. Setting up the complete algorithm is now straightforward and is presented in Algorithm 4.5.

Remark 4.3
Although it is possible to formulate a minimisation strategy with the Bregman iteration for L_1–L_1 models, the formulation appears a bit "unnatural". Three potential problems become immediately apparent. The first one being the fact that we had to eliminate the variables, with respect to which we initially wanted to minimise, completely from the cost function. Secondly, none of the model parameters has a direct influence on u and v. They can only interact by means of the auxiliary variables. Chances are, that this will reduce the responsiveness of the algorithm to parameter changes. Although it is generally desirable to have algorithms that do not react too sensitive with respect to varying parameters, the other extreme of having an algorithm that reacts hardly at all, is not desirable as well. The third point is that the alternating minimisation scheme contains many more steps compared to all the previous models. It might take a lot of iterations until we are close to a minimum. These observations certainly raise a certain number of concerns about the efficiency of this algorithm.

Remark 4.4
A rotationally invariant formulation is also possible. Basically one has to do the same adaptations to this model as we had to do in Section 4.7 for the L_2–L_1 model. Again, the only significant change will lie in the minimisation of the auxiliary variables.

4.9 Summary of the results from the previous sections

Sections 4.4 to 4.8 represent the most important parts of this thesis. We have seen in these sections how the Bregman framework can be applied to the discretised formulations of the energy functionals given in Section 4.1.3 and we have presented algorithms based upon the split Bregman formulation that allow us to find minimisers of each model considered in that section. Basically, all the discretised models were of the form

$$\arg\min_{\eta} \sum_{i=1}^{N} \|A_i \eta - b_i\| + \sum_{j=1}^{M} \|B_j \eta - c_j\| \tag{4.79}$$

Data: F, F_x, F_y, $\boldsymbol{f_t}$, $\boldsymbol{f_{xt}}$, $\boldsymbol{f_{yt}}$, λ, γ, μ, N, M
Result: u^N and v^N minimising eq. (4.73)
Initialize: $u^0 = v^0 = 0$, $d^0 = d_x^0 = d_y^0 = 0$, $\boldsymbol{d^u}^0 = \boldsymbol{d^v}^0 = 0$, ${\boldsymbol{f_t}}^0 = \boldsymbol{f_t}$, $= {\boldsymbol{f_{xt}}}^0 = \boldsymbol{f_{xt}}$, ${\boldsymbol{f_{yt}}}^0 = \boldsymbol{f_{yt}}$, $\boldsymbol{b^u}^{,0} = \boldsymbol{b^v}^{,0} = 0$
for $i = 0$ **to** $N - 1$ **do**
  $u^{i,0} = u^i$
  $v^{i,0} = v^i$
  $d^{i,0} = d^i$, $d_x^{i,0} = d_x^i$, $d_y^{i,0} = d_y^i$
  $\boldsymbol{d^u}^{,i,0} = \boldsymbol{d^u}^{,i}$, $\boldsymbol{d^v}^{,i,0} = \boldsymbol{d^v}^{,i}$
  for $j = 0$ **to** $M - 1$ **do**
    Solve

$$\begin{aligned}\left(u^{i,j+1}, v^{i,j+1}\right) = \arg\min_{u,v} &\left\| d^{i,j} - F\binom{u}{v} - b^i \right\|_2^2 \\ &+ \left\| d_x^{i,j} - F_x\binom{u}{v} - b_x^i \right\|_2^2 + \left\| d_y^{i,j} - F_y\binom{u}{v} - b_y^i \right\|_2^2 \\ &+ \frac{\mu}{2}\left(\left\| \boldsymbol{d^u}^{,i,j} - \boldsymbol{\nabla}\, u - \boldsymbol{b^u}^{,i} \right\|_2^2 + \left\| \boldsymbol{d^v}^{,i,j} - \boldsymbol{\nabla}\, v - \boldsymbol{b^v}^{,i} \right\|_2^2 \right)\end{aligned}$$

    by finding a solution of eq. (4.78)

$$d^{i,j+1} = \text{shrink}\left(D_x u^{i,j+1} + D_y v^{i,j+1} + {\boldsymbol{f_t}}^i, \frac{\lambda}{\mu} \right) \text{ (done componentwise)}$$

$$d_x^{i,j+1} = \text{shrink}\left(D_{xx} u^{i,j+1} + D_{xy} v^{i,j+1} + {\boldsymbol{f_{xt}}}^i, \frac{\lambda\gamma}{\mu} \right) \text{ (done componentwise)}$$

$$d_y^{i,j+1} = \text{shrink}\left(D_{yx} u^{i,j+1} + D_{yy} v^{i,j+1} + {\boldsymbol{f_{yt}}}^i, \frac{\lambda\gamma}{\mu} \right) \text{ (done componentwise)}$$

$$d_{k,l}^{u,i,j+1} = \text{gshrink}\left(\nabla u_{k,l}^{i,j+1} + b_{k,l}^{u,i}, \frac{1}{\mu} \right) \ \forall k, l$$

$$d_{k,l}^{v,i,j+1} = \text{gshrink}\left(\nabla v_{k,l}^{i,j+1} + b_{k,l}^{v,i}, \frac{1}{\mu} \right) \ \forall k, l$$

  end
  $u^{i+1} = u^{i,M}$, $v^{i+1} = v^{i,M}$
  $d^{i+1} = d^{i,M}$, $d_x^{i+1} = d_x^{i,M}$, $d_y^{i+1} = d_y^{i,M}$
  $\boldsymbol{d^u}^{,i+1} = \boldsymbol{d^u}^{,i,M}$, $\boldsymbol{d^v}^{,i+1} = \boldsymbol{d^v}^{,i,M}$
  ${\boldsymbol{f_t}}^{i+1} = {\boldsymbol{f_t}}^i + \boldsymbol{f_t} - d^{i+1} + D_x u^{i+1} + D_y v^{i+1}$
  ${\boldsymbol{f_{xt}}}^{i+1} = {\boldsymbol{f_{xt}}}^i + \boldsymbol{f_{xt}} - d_x^{i+1} + D_{xx} u^{i+1} + D_{xy} v^{i+1}$
  ${\boldsymbol{f_{yt}}}^{i+1} = {\boldsymbol{f_{yt}}}^i + \boldsymbol{f_{yt}} - d_y^{i+1} + D_{yx} u^{i+1} + D_{yy} v^{i+1}$
  $\boldsymbol{b^u}^{,i+1} = \boldsymbol{b^u}^{,i} - \boldsymbol{d^u}^{,i+1} + \boldsymbol{\nabla}\, u^{i+1}$
  $\boldsymbol{b^v}^{,i+1} = \boldsymbol{b^v}^{,i} - \boldsymbol{d^v}^{,i+1} + \boldsymbol{\nabla}\, v^{i+1}$
end

Algorithm 4.5: The split Bregman algorithm for the non-rotationally invariant L_1–L_1 model.

with linear operators A_i and B_j, vectors b_i and c_j and different choices for the norms. It is interesting to note that all our optical flow algorithms are very similar. They consist essentially of a linear system and shrinkage operations, independent of the choice of the data and smoothness term. Minimising an energy functional with a robust data term and/or a robust smoothness term is hardly more complicated than minimising the rather simple model of Horn and Schunck. In this context it is important to emphasize that each algorithm is in fact simply an application of the split Bregman iteration as formulated in Algorithm 3.4. We showed in Proposition 3.8, Proposition 3.11 and Proposition 3.13 that the Bregman algorithm and thus also the split Bregman formulation converge. Therefore, it follows that the Algorithms 4.1 to 4.5 yield minimisers of the six functionals presented in Section 4.1.5. This observation underlines the importance of the detailed presentation of the Bregman framework given in Chapter 3. It guarantees us that our algorithms will produce the desired displacement fields.

4.10 Properties of the linear systems occurring in the Bregman algorithms

All the linear systems that appeared in our algorithms so far were of the form

$$\begin{aligned}(D_xD_x + \gamma D_{xx}D_{xx} + \gamma D_{yx}D_{yx})\,u + (D_xD_y + \gamma D_{xx}D_{xy} + \gamma D_{yx}D_{yy})\,v \\ - \theta\boldsymbol{\Delta}u = R_u \\ (D_xD_y + \gamma D_{xx}D_{xy} + \gamma D_{yx}D_{yy})\,u + (D_yD_y + \gamma D_{xy}D_{xy} + \gamma D_{yy}D_{yy})\,v \\ - \theta\boldsymbol{\Delta}v = R_v\end{aligned} \tag{4.80}$$

with parameters $\gamma > 0$, $\theta > 0$ and righthand side vectors R_u and R_v. If n_p is the total number of pixels, then this system has $2n_p$ equations and $2n_p$ unknowns. It is interesting to note that the discretisation of the Euler-Lagrange equations of the Horn and Schunck model would lead to a linear system with almost the same structure. See for example [20, 57]. In [57] the authors analysed this linear system and showed that the discretisation of the Euler-Lagrange equations leads to symmetric and positive definite matrix. Because of the high similarity between the two problems it will be relatively simple to adapt their proof such that we can show the same results for our Bregman algorithms. We will even demonstrate that the proof given in [57] can be generalised. The authors of this article required a specific indexing scheme for the pixels and assumed that there was only one constancy assumption, namely the gray value constancy. The proof given in this section demonstrates that these assumptions are not necessary. We will show that the inclusion of higher order constancy assumptions does not affect the positive definiteness.

The fact that the matrix is symmetric and positive definite is highly useful for numerical purposes. It guarantees the convergence of algorithms such as conjugate gradients and will allow us later on to present efficient implementations with powerful solvers.

For the sake of simplicity, we will assume in the following that our image is discretized on a rectangular grid with step sizes h_x and h_y in the respective direction. We will further assume that the pixels are indexed by a single number $i \in \{1, \dots, n_p\}$. The neighbouring

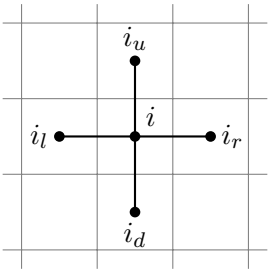


Figure 4.2: Pixel naming convention. $N_x(i) = \{i_l, i_r\}$ whereas $N_y(i) = \{i_u, i_d\}$.

pixels will be labelled i_l, i_r, i_u and i_d, where the indices stand for *left*, *right*, *up* and *down*. The sets $N_x(i)$ and $N_y(i)$ will represent the neighbours of pixel i in x (resp. y) direction. Figure 4.2 summarises the naming convention presented in this paragraph.

It is easy to see that the system matrix given in eq. (4.80) is symmetric and positive semi-definite. This fact follows from the observation that for an arbitrary matrix

$$0 \leqslant \|Ax\|_2^2 = x^T A^T A x \tag{4.81}$$

and thus $A^T A$ is symmetric and positive semi-definite. The linear system in eq. (4.80) was obtained by differentiating terms of the form $\left\|A\binom{u}{v} + b\right\|_2^2$ for some matrix A and vector b. Thus, it must also have these properties. However, proving that the matrix is even positive definite will not be as easy. If we call the matrix corresponding to eq. (4.80) M, then we must show that it fulfils $\binom{u}{v}^T M \binom{u}{v} > 0$ for all $\binom{u}{v} \neq 0$.

The first step, that we will perform, will be to rewrite the considered system in a more explicit form. The matrices D_x, D_y, etc. are all diagonal matrices, thus it follows that they can easily be multiplied with each other. As for $\mathbf{\Delta}$, we will assume that the second derivatives are approximated in the following way

$$\begin{aligned} (\partial_{xx} u)_i &\approx \frac{u_{i_l} - 2u_i + u_{i_r}}{h_x^2} \\ (\partial_{yy} u)_i &\approx \frac{u_{i_u} - 2u_i + u_{i_d}}{h_y^2} \end{aligned} \tag{4.82}$$

and therefore,

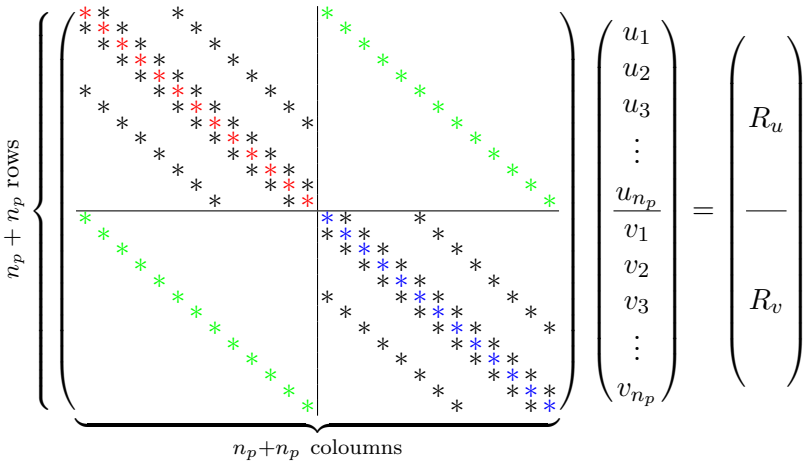
$$(\mathbf{\Delta} u)_i = (\partial_{xx} u)_i + (\partial_{yy} u)_i = \sum_{k \in N_x(i)} \frac{u_k - u_i}{h_x^2} + \sum_{k \in N_y(i)} \frac{u_k - u_i}{h_y^2} \tag{4.83}$$

Of course, the same formula also holds for $(\mathbf{\Delta} v)_i$. This leads us to the following explicit form of our linear system ($i = 1, \ldots, n_p$)

$$\begin{aligned} \left(f_x{}^2 + \gamma\left(f_{xx}^2 + f_{xy}^2\right)\right)_i u_i + \left(f_x f_y + \gamma\left(f_{xx} f_{xy} + f_{xy} f_{yy}\right)\right)_i v_i \\ - \theta \sum_{k \in N_x(i)} \frac{u_k - u_i}{h_x^2} - \theta \sum_{k \in N_y(i)} \frac{u_k - u_i}{h_y^2} = (R_u)_i \end{aligned} \tag{4.84}$$

$$\begin{aligned}(f_x f_y + \gamma\left(f_{xx} f_{xy} + f_{xy} f_{yy}\right))_i u_i + \left(f_y^2 + \gamma\left(f_{xy}^2 + f_{yy}^2\right)\right)_i v_i \\ - \theta \sum_{k \in N_x(i)} \frac{v_k - v_i}{h_x^2} - \theta \sum_{k \in N_y(i)} \frac{v_k - v_i}{h_y^2} = (R_v)_i\end{aligned} \tag{4.85}$$

In eq. (4.80), the system is written down with matrices. Thus, the first equation of eq. (4.80) corresponds to the n_p equations given by eq. (4.84), whereas the second equation of eq. (4.80) corresponds to the n_p equations given by eq. (4.85). If we had numbered the equations consecutively, then eq. (4.84) would correspond to the equations 1 to n_p and eq. (4.85) would correspond to the equations $n_p + 1$ to $2n_p$. However, because of the special structure of these equations, it is usually more convenient to write them down pairwise. Equation (4.86) shows the structure of the system matrix if the pixels had been numbered row-wise. With the definitions of eq. (4.87), the red marks contain terms from J_{11} and $\mathbf{\Delta} u$, the blue marks correspond to terms stemming from J_{22} and $\mathbf{\Delta} v$ and the green marks correspond to J_{12}. The remaining marks contain terms from $\mathbf{\Delta}$.



(4.86)

If we define the abbreviations

$$\begin{aligned}(J_{11})_i &:= \left(f_x{}^2 + \gamma\left(f_{xx}^2 + f_{xy}^2\right)\right)_i \\ (J_{12})_i &:= \left(f_x f_y + \gamma\left(f_{xx} f_{xy} + f_{xy} f_{yy}\right)\right)_i \\ (J_{22})_i &:= \left(f_y^2 + \gamma\left(f_{xy}^2 + f_{yy}^2\right)\right)_i\end{aligned} \tag{4.87}$$

then we obtain the following final form (again $i = 1, \ldots, n_p$)

$$(J_{11})_i\, u_i + (J_{12})_i\, v_i - \theta \sum_{k \in N_x(i)} \frac{u_k - u_i}{h_x^2} - \theta \sum_{k \in N_y(i)} \frac{u_k - u_i}{h_y^2} = (R_u)_i \tag{4.88}$$

$$(J_{12})_i\, u_i + (J_{22})_i\, v_i - \theta \sum_{k \in N_x(i)} \frac{v_k - v_i}{h_x^2} - \theta \sum_{k \in N_y(i)} \frac{v_k - v_i}{h_y^2} = (R_v)_i \tag{4.89}$$

By using this representation of the linear system, $0 < \binom{u}{v}^T M \binom{u}{v}$ can be rewritten as

$$0 < \sum_{i=1}^{n_p} u_i \left[(J_{11})_i u_i + (J_{12})_i v_i - \theta \sum_{k \in N_x(i)} \frac{u_k - u_i}{h_x^2} - \theta \sum_{k \in N_y(i)} \frac{u_k - u_i}{h_y^2} \right] \tag{4.90}$$
$$+ \sum_{i=1}^{n_p} v_i \left[(J_{12})_i u_i + (J_{22})_i v_i - \theta \sum_{k \in N_x(i)} \frac{v_k - v_i}{h_x^2} - \theta \sum_{k \in N_y(i)} \frac{v_k - v_i}{h_y^2} \right]$$

Regrouping the terms we obtain

$$\sum_{i=1}^{n_p} \left[(J_{11})_i u_i^2 + 2 (J_{12})_i u_i v_i + (J_{22})_i v_i^2 - \theta u_i \sum_{k \in N_x(i)} \frac{u_k - u_i}{h_x^2} \right. \tag{4.91}$$
$$\left. - \theta u_i \sum_{k \in N_y(i)} \frac{u_k - u_i}{h_y^2} - \theta v_i \sum_{k \in N_x(i)} \frac{v_k - v_i}{h_x^2} - \theta v_i \sum_{k \in N_y(i)} \frac{v_k - v_i}{h_y^2} \right]$$

By applying the definitions of J_{11}, J_{12} and J_{22} it is easy to see that the first three terms in each addend can be rewritten as

$$(f_x u + f_y v)_i^2 + \gamma \left[(f_{xx} u + f_{xy} v)_i^2 + (f_{xy} u + f_{yy} v)_i^2 \right] \tag{4.92}$$

and thus they are always non-negative. Let us now consider the remaining terms (omitting θ as it is strictly positive anyway).

$$\sum_{i=1}^{n_p} \left[\sum_{k \in N_x(i)} \frac{u_i^2 - u_i u_k}{h_x^2} + \sum_{k \in N_y(i)} \frac{u_i^2 - u_i u_k}{h_y^2} + \sum_{k \in N_x(i)} \frac{v_i^2 - v_i v_k}{h_x^2} + \sum_{k \in N_y(i)} \frac{v_i^2 - v_i v_k}{h_y^2} \right] \tag{4.93}$$

In order to show that eq. (4.93) is also positive, we will have to reorder these terms one more time. Assume that we are in pixel i and that this pixel has a neighbour in every direction. (If not, then certain terms in the following reflection are simply not present). Then we preform the following exchanges (names are always based on the point of view of i):

- Pixel i receives the terms $\frac{1}{h_x^2} \left(u_{i_r}^2 - u_i u_{i_r} \right)$ from pixel i_r and $\frac{1}{h_y^2} \left(u_{i_d}^2 - u_i u_{i_d} \right)$ from pixel i_d.
- Pixel i receives the terms $\frac{1}{h_x^2} \left(v_{i_r}^2 - v_i v_{i_r} \right)$ from pixel i_r and $\frac{1}{h_y^2} \left(v_{i_d}^2 - v_i v_{i_d} \right)$ from pixel i_d.
- Pixel i gives the terms $\frac{1}{h_x^2} \left(u_i^2 - u_i u_{i_l} \right)$ to pixel i_l and $\frac{1}{h_x^2} \left(u_i^2 - u_i u_{i_u} \right)$ to pixel i_u.
- Pixel i gives the terms $\frac{1}{h_x^2} \left(v_i^2 - v_i v_{i_l} \right)$ to pixel i_l and $\frac{1}{h_x^2} \left(v_i^2 - v_i v_{i_u} \right)$ to pixel i_u.

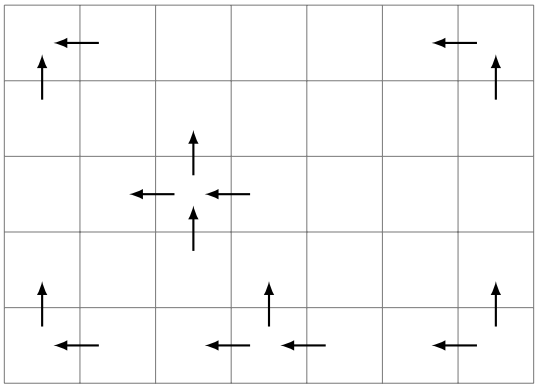


Figure 4.3: Visualisation of the reordering scheme for the pixels of the image. Arrows depict to which pixels the different terms are reassigned.

Figure 4.3 visualises the idea behind this reordering. The arrows depict the direction in which a term is moved. It follows now that eq. (4.93) can be rewritten as

$$\sum_{i=1}^{n_p}\left[\sum_{k\in\{i_r\}\cap N_x(i)}\frac{1}{h_x^2}\left(u_i^2+u_k^2-2u_iu_k\right)+\sum_{k\in\{i_d\}\cap N_y(i)}\frac{1}{h_y^2}\left(u_i^2+u_k^2-2u_iu_k\right)+\right. \tag{4.94}$$
$$\left.\sum_{k\in\{i_r\}\cap N_x(i)}\frac{1}{h_x^2}\left(v_i^2+v_k^2-2v_iv_k\right)+\sum_{k\in\{i_d\}\cap N_y(i)}\frac{1}{h_y^2}\left(v_i^2+v_k^2-2v_iv_k\right)\right]$$

which conveniently simplifies to

$$\sum_{i=1}^{n_p}\left[\sum_{k\in\{i_r\}\cap N_x(i)}\frac{1}{h_x^2}\left(u_i-u_k\right)^2+\sum_{k\in\{i_d\}\cap N_y(i)}\frac{1}{h_y^2}\left(u_i-u_k\right)^2+\right. \tag{4.95}$$
$$\left.\sum_{k\in\{i_r\}\cap N_x(i)}\frac{1}{h_x^2}\left(v_i-v_k\right)^2+\sum_{k\in\{i_d\}\cap N_y(i)}\frac{1}{h_y^2}\left(v_i-v_k\right)^2\right]$$

Equation (4.95) is 0 if and only if $u_i = \text{const}_u$ and $v_i = \text{const}_v$ for all i. But then, it follows from eq. (4.92) that $(f_xu + f_yv)_i^2 = 0$ can only be verified for all i if and only if the spatial gradient ∇f is perpendicular to the flow field (u, v). In particular, it has to be constant as well. On the other hand, ∇f is also perpendicular to the level curves $L_c := \{x \mid f(x,t) = c\}$. This implies, that the flow field must be tangent to L_c at every point. All in all, this would mean that in the continuous setting the graph of f would have to be a plane in $\mathbb{R}^3$ for all times t. Thus, if we exclude this case, where the graph is a plane, then the system matrix that we obtain in our Bregman iterations is always positive definite.

Remark 4.5
The above argumentation also holds if we only consider the gray value constancy, i.e. $\gamma = 0$. On the other hand, the gray value constancy cannot be removed. If we only

considered the constancy of the gradient, then the matrix is not necessarily positive definite.

4.11 Handling large displacements with coarse-to-fine strategies

All the data terms in our models use linearised constancy assumptions. The linearisation is essential to obtain convex energy functionals which allow us to apply the Bregman iterations. However, it restrains our algorithms to image sequences with small displacements. To also handle modern sequences with large displacements, we follow [18, 20, 78] and embed the minimisation of our energy into a coarse-to-fine multiscale warping approach using downsampled images. This means that we consider our image sequence at different resolution levels. On the coarsest level the displacement will be small enough to justify the models with linearised constancy assumptions; thus we can compute the displacement field with our algorithms from the previous sections. On the subsequent finer levels we split the flow field into a known part (u^c, v^c) which consists of the solution from the coarser level that we projected through interpolation to the next finer level and an unknown small flow increment $(\mathrm{d}u, \mathrm{d}v)$. Then, in a so called warping step, we compute $f(x + u^c, y + v^c, t + 1)$ by compensating the second frame $f(x, y, t + 1)$ by (u^c, v^c). In order give a well understandable presentation of this warping step we will now assume that we operate on a discrete grid with step size 1 in each direction. Extensions to arbitrary grid sizes are simply a matter of scaling, but they would render the notations cumbersome. In general, the flow components (u^c, v^c) are not multiples of the grid size. Therefore, we perform the compensation by splitting (u^c, v^c) into an an integer part $(\bar{u}, \bar{v})$ and a fractional part $(\varepsilon_u, \varepsilon_v)$:

$$\begin{aligned} u^c &= \bar{u} + \varepsilon_u \\ v^c &= \bar{v} + \varepsilon_v \end{aligned} \tag{4.96}$$

Then we compute the value at $f(x + u^c, y + v^c, t + 1)$ through bilinear interpolation of the surrounding pixels.

$$\begin{aligned} [f(x + u^c, y + v^c, t + 1)]_{i,j} = {} & (1 - \varepsilon_u)(1 - \varepsilon_v)\,[f(x, y, t + 1)]_{i+\bar{u},j+\bar{v},t+1} \\ & + (\quad \varepsilon_u)(1 - \varepsilon_v)\,[f(x, y, t + 1)]_{i+\bar{u}+1,j+\bar{v},t+1} \\ & + (1 - \varepsilon_u)(\quad \varepsilon_v)\,[f(x, y, t + 1)]_{i+\bar{u},j+\bar{v}+1,t+1} \\ & + (\quad \varepsilon_u)(\quad \varepsilon_v)\,[f(x, y, t + 1)]_{i+\bar{u}+1,j+\bar{v}+1,t+1} \end{aligned} \tag{4.97}$$

Proceeding with this formula for every pixel we obtain the warped image - the modified version of the image at time $t+1$ with respect to the displacement vector field. Figure 4.4 also visualises the idea behind this warping. Note that the warping step incorporates the current solution into the problem. The image at time $t + 1$ is warped successively closer to the image at time t. Therefore, what remains to be computed at each level is the difference problem, i.e the next finer motion increment $(\mathrm{d}u, \mathrm{d}v)$ such that

$$f(x + u^c + \mathrm{d}u, y + v^c + \mathrm{d}v, t + 1) = f(x, y, t) \tag{4.98}$$

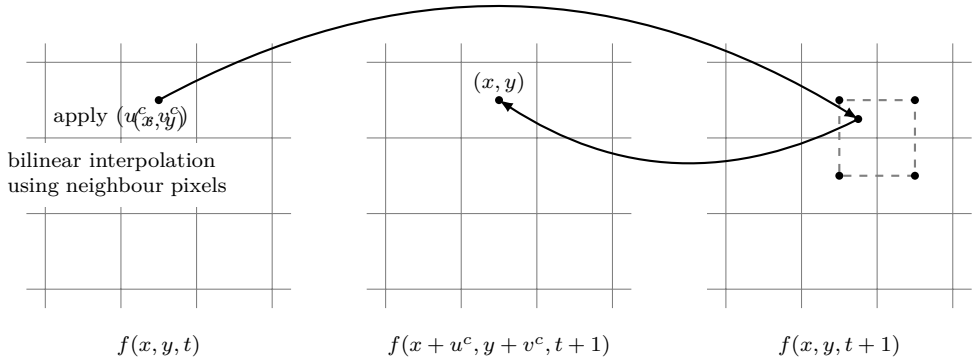


Figure 4.4: Visualisation of the warping step. The four pixels linked with a dashed line mark the points used for the bilinear interpolation.

Since the increments are supposed to be small, the constancy assumptions can again be linearised (with respect to du and dv) and one proceeds as before. After having obtained the increments, one projects everything to the next finer resolution level and repeats the procedure. In the end, one obtains the complete flow field by summing up all the increments from the different levels. In this thesis we will always perform the resampling by using an area averaging strategy as detailed in [21, 78], but it should also be clear that any reasonable interpolation strategy can be applied.

Reasonable downsampling factors for the image resolution of such a coarse-to-fine approach lie in the range $[0.5, 1)$. Larger factors are likely to yield better approaches as they provide smoother transitions from one level to the next. However, they also increase the computational burden because one has more different resolution levels to compute the flow field at.

4.11.1 Further enhancements

Our optical flow models all had a regularisation parameter λ, either placed in front of the data term or in front of the smoothness term. The role of this parameter was to dictate how smooth our solution should be. The idea is now to use a different regularisation weight at each resolution level. If the resolution is coarse, then the resulting displacement vector field should be very smooth. On finer levels it should be less smooth, thus allowing discontinuities in the flow field caused by small objects that did not appear on the coarser levels. One possible choice would be to set the regularisation weight $\alpha(l)$ at level l (0 being the finest level and $N > 0$ the coarsest) to $\alpha(l) := \lambda c^l$. The positive constant c should be larger than 1 if the regularisation parameter is placed in front of the smoothness term and smaller than 1 if the regularisation weight stands in front of the data term. This guarantees that we get smoother results on coarser levels. Another benefit from this approach is that it stabilizes the interpolation steps that we perform during the coarse-to-fine scheme.

Additionally to the above mentioned strategy, we follow an idea from [86]. There, the authors suggested to apply a median filter on the components (u^c, v^c) that were obtained

from the coarser grid. This eliminates strong outliers that could lead to erroneous results in the warping step at the next finer level.

4.12 Occlusion detection

Some of the models that we presented in Section 4.1.3 contain quadratic data terms and thus, they impose a harsh penalisation on outliers. As occluded pixels cannot fulfill any constancy assumptions, it makes sense to disable the data term at occlusions in order to stabilise the algorithm and improve the results. This can be achieved by multiplying the data term with an occlusion indicator function $o \in \{0, 1\}$, where $o(x, y) = 0$ if a pixel is occluded and $o(x, y) = 1$ if a pixel is visible. This way, the flow of field of occluded pixels is solely determined by the smoothness term and not by the erroneous information of the data term.

There are several possibilities to detect occlusions. We will follow the popular cross-checking technique from [34, 67]. The idea is the following: one determines, in addition to the (forward) flow field (u, v), a backward flow field (u_b, v_b) that describes the displacements from time $t + 1$ to t. It is clear, that for non occluded pixels, the forward flow will be the inverse of the backward flow. This motivates the definition of the cross-check value by

$$c(x, y) := \left\| \begin{pmatrix} u(x, y) + u_b(x + u(x, y), y + v(x, y)) \\ v(x, y) + v_b(x + u(x, y), y + v(x, y)) \end{pmatrix} \right\|_2 \tag{4.99}$$

If (x, y) is the location of an occluded pixel, then $c\,(x, y)$ will be large, since the forward and backward flow will not cancel each other. For non-occluded pixels, $c\,(x, y)$ will be close to 0. Thus, we obtain the occlusion indicator function $o(x, y)$ by thresholding $c(x, y)$ using a threshold parameter T_{occ}.

Note that this strategy requires that the flow field needs to be computed at least three times. Twice two determine the forward and backward flow and a third time where we apply the occlusion indicator function. Thus, this strategy triples the computation time and should only be applied on sequences that require such a proper occlusion handling.

4.13 Summary and concluding remarks

In this chapter we have seen how the split Bregman algorithm can be applied to optical flow problems. We presented a certain number of models based on variational formulations and showed how they can be discretized and solved with the split Bregman algorithm. The corresponding algorithms were then presented in Sections 4.4 to 4.8. We also discussed a certain number of possible improvements like occlusion handling (Section 4.12) and coarse-to-fine strategies (Section 4.11) that help us tackle some of the weaknesses of the optical flow models. The occlusion handling is especially important for those approaches with a quadratic data term, whereas the coarse-to-fine strategy helps us to overcome the restriction that the displacements must be small. As we could see, the formulations of all the algorithms are very similar. One Bregman iteration always consists in solving

linear systems and applying thresholding operations. It follows that these algorithms are generally relatively easy to implement. However, we also noticed that not every model is equally well suited for the Bregman framework. The model of Horn and Schunck is "too simple" and can be handled much more efficiently with the Euler-Lagrange equations. The L_1–L_1 models can be treated with the Bregman iterations but also in this case there appear to exist hints that the convergence behaviour might be suboptimal. The remaining models with one differentiable and one non-differentiable term fit naturally into the Bregman framework and are likely to yield the best performance. Finally, one should also mention the results from Section 4.10. The positive definiteness of the system matrix allows us to consider a broad range of highly efficient algorithms for solving the occuring linear systems.

5 Numerical evaluation

The goal of this chapter will be to demonstrate the usefulness of the Bregman framework by applying some of our algorithms from Chapter 4 on a certain number of test sequences. We will use the data sets from the Middlebury computer vision page [74]. The correct ground truth of these sequences is known; therefore, it allows us to present an accurate evaluation of our algorithms. In order to give a quantitative representation of the accuracy of the obtained flow fields, we will consider the so called *average angular error* given by

$$\mathrm{AAE}\left(u_e, u_c\right) := \frac{1}{n_p} \sum_{i,j} \arccos \frac{\langle u_{ci,j}, u_{ei,j} \rangle}{\left\|u_{ci,j}\right\|_2 \left\|u_{ei,j}\right\|_2} \tag{5.1}$$

as well as the *average endpoint error* defined as

$$\mathrm{AEE}\left(u_e, u_c\right) := \frac{1}{n_p} \sum_{i,j} \left\|u_{ci,j} - u_{ei,j}\right\|_2 \tag{5.2}$$

The subscripts c and e denote the correct respectively the estimated spatiotemporal optic flow vectors $u_c = (u_{c1}, u_{c2}, 1)^T$ and $u_e = (u_{e1}, u_{e2}, 1)^T$. In this context n_p denotes the number of pixels of an image from the considered sequence. The average angular error is a relatively popular method to measure the quality of a flow field. However, as mentioned by the authors of [6, 7], it is slightly biased. Errors in large flows are less penalised than errors in small flows. In order to offer a fair comparison, we will, therefore, also consider the endpoint error as suggested in [6].

As for the qualitative evaluation of the computed flow fields, we will use the colour representation shown in Fig. 5.1. Here, the hue encodes the direction and the brightness represents the magnitude of the vector. Although the color coding is difficult to interpret exactly, it provides a comfortable way to visualise, at the same time, both direction and magnitude of the displacement field.

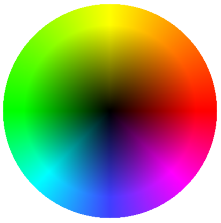
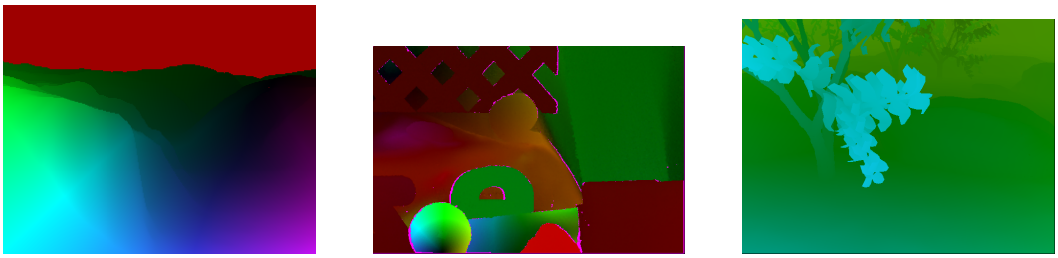


Figure 5.1: Color code for the displacement field.

(a) *Yosemite* sequence (b) *Rubberwhale* sequence (c) *Grove 2* sequence



(d) *Yosemite* ground truth (e) *Rubberwhale* ground truth (f) *Grove 2* ground truth

Figure 5.2: **First Row:** A frame from the tested sequences. **Second Row:** Corresponding exact ground truth. Magenta pixels in the exact flow field depict locations where the correct ground truth cannot be given.

5.1 Evaluation of the Bregman algorithms

In the following we will use the sequences depicted in Fig. 5.2 to test our algorithms. From the five algorithms that we presented in Chapter 4 we will actually use the L_1–L_2 algorithm from Section 4.5, the rotationally invariant L_2–L_1 algorithm from Section 4.7 and the rotationally invariant L_1–L_1 approach mentioned in Section 4.8. Each algorithm was implemented as described in Chapter 4. The occurring linear system was always solved with a simple Gauß-Seidel algorithm. Additionally, we added a coarse-to-fine strategy as described in Section 4.11 and set the downsampling factor in all the tests to 0.9. The strategy with the median filter from Section 4.11.1 was also implemented. The regularisation parameter, however, was left constant on each warping level. The number of iterations for each model was chosen in such a way that the algorithm reached convergence for every considered sequence.

Figure 5.3 depicts the obtained flow fields and the tables 5.1 to 5.3 present the parameter choices as well as the error measures and runtimes for the different algorithms. In this context AAE denotes the average angular error, AEE the average endpoint error and RT the runtime in seconds. The meanings of the parameters λ, μ and γ are the same as in the descriptions of the algorithms in Chapter 4. σ is the standard deviation used for the preprocessing of the images with a gaussian convolution. Note that the values of the parameter λ vary significantly in Table 5.2. The reason for these differences lies in the fact that we placed λ in front of the smoothness term for the L_1-L_2 model and in front of the data term for the other models. Concerning the results, it is interesting to note

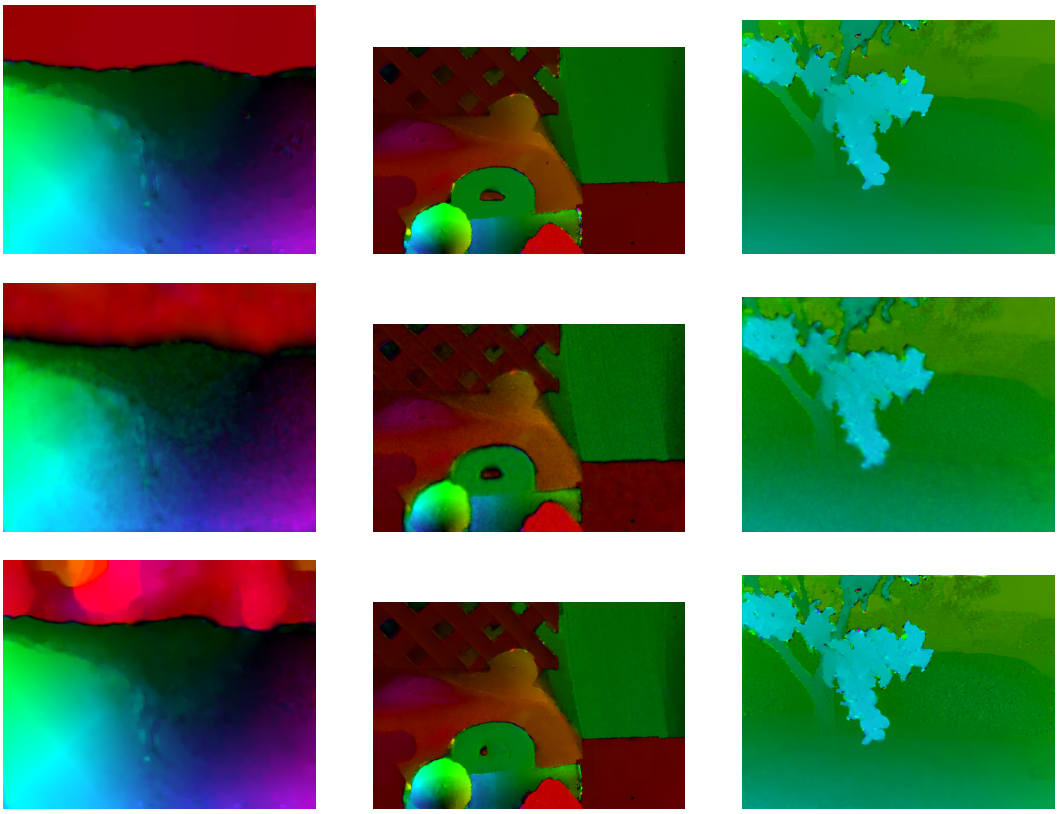


Figure 5.3: The resulting flow fields. **First row:** L_2–L_1 model. **Second row:** L_1–L_2 model. **Third row:** L_1–L_1 model.

Sequence	λ	μ	γ	σ	AAE	AEE	RT
Yosemite	0.0025	8.45	84.50	1.30	2.91	0.12	35
Rubberwhale	0.0100	11.25	20.00	0.40	4.06	0.12	93
Grove 2	0.0250	6.30	1.50	0.75	2.79	0.18	125

Table 5.1: Parameter choices and errors for the L_2–L_1 model with 30 Bregman iterations, 10 Gauss-Seidel iterations and 3 alternating minimisations.

Sequence	λ	μ	γ	σ	AAE	AEE	RT
Yosemite	1300	1.10	15.00	0.90	4.57	0.19	72
Rubberwhale	1125	8.45	23.00	0.40	5.79	0.17	180
Grove 2	475	4.75	4.75	0.80	3.48	0.23	242

Table 5.2: Parameter choices and errors for the L_1–L_2 model with 50 Bregman iterations, 10 Gauss-Seidel iterations and 3 alternating minimisations.

Sequence	λ	μ	γ	σ	AAE	AEE	RT
Yosemite	0.0040	0.13	47.50	1.60	5.17	0.28	215
Rubberwhale	0.0065	0.23	1.00	0.38	4.67	0.14	530
Grove 2	0.0650	0.41	1.00	0.90	2.95	0.20	720

Table 5.3: Parameter choices and errors for the L_1–L_1 model with 150 Bregman iterations, 10 Gauss-Seidel iterations and 3 alternating minimisations.

that the L_2–L_1 model converged much faster than the L_1–L_1 formulation. This is likely due to the fact that in the L_1–L_1 algorithm we placed all the terms from the original energy functional into the constraining condition, whereas for the other models, this was not the case.

5.2 Comparison with other algorithms

We already presented various numerical evaluations for our Bregman formulations in the previous section. In general, these results seem to be rather good. However, we do not know yet how our approach compares to state of the art algorithms from the literature. This will be subject of the next two sections. We will compare two of our Bregman iterations from Chapter 4 with other algorithms that minimise the same variational model. All our tests will be done using the *Rubberwhale* and *Dimetrodon* sequences from the Middlebury pages [74].

5.2.1 Bregman L_1-L_1 without gradient constancy and TV-L_1-M

In the following we will consider the TV-L_1 method [95] with additional median filtering [86] (TV-L_1-M) and compare it to the anisotropic L_1-L_1 Bregman formulation from Section 4.8. The TV-L_1-M approach entails a similar strategy as the Bregman framework, in the sense, that it introduces an additional variable and splits the minimisation process into two subproblems. One of them is solved with shrinkage operators, whereas the other requires the use of Chambolle's algorithm [30]. Our implementation of the TV-L_1-M formulation will correspond to the presentation from [95]. In order to guarantee a fair comparison, we will, however, use the same warping scheme for both methods and

only consider the grey value constancy for the Bregman approach, i.e. we set $\gamma = 0$ in Section 4.8. Under these conditions the considered algorithms solve exactly the same model. Evaluating both approaches confirms the good results of the Bregman framework from the previous section. The L_1-L_1 formulation achieves an average endpoint error of 0.15 for the *Rubberwhale* sequence and 0.13 for the *Dimetrodon* sequence. On the other hand, the TV-L_1-M algorithm returns an endpoint error of 0.16 for *Rubberwhale* and 0.15 for *Dimetrodon*. However, one should also note that the computation time of the Bregman algorithm was 600 seconds for the *Rubberwhale* sequence and 140 seconds for the *Dimetrodon* sequence, whereas the TV-L_1-M formulation only needed 200 seconds for *Rubberwhale* and 40 seconds for *Dimetrodon*. It follows that our algorithm is competitive in terms of accuracy but not in terms of speed. The parameters that were used for the evaluation are described in Table 5.4 and Table 5.5. The meanings of Bregman parameters are the same as in the previous section. The notations in Table 5.5 correspond to those in the references [86, 95]. The only parameter that is not mentioned in these tables is the downsampling factor of the warping scheme. Unless mentioned differently, it is always set to 0.9.

5.2.2 Bregman L_2-L_1 and Brox-QDT-M

In this section we compare the isotropic Bregman L_2-L_1 formulation from Section 4.7 to a variant of the method of Brox et al. [18] with a quadratic data term and median filtering (Brox-QDT-M). This yields a model that only differs from ours by the differentiable approximation of the TV regulariser required by the Euler-Lagrange framework. The split Bregman algorithm was specifically designed to handle terms like $\|\nabla u\|_2$. However, the Euler-Lagrange equations require differentiable terms. Therefore, one introduces the following approximation

$$\|\nabla u\|_2 \approx \sqrt{(\partial_x u)^2 + (\partial_y u)^2 + \varepsilon^2} \tag{5.3}$$

with a small $\varepsilon > 0$. The Brox-QDT-M algorithm also uses a multigrid scheme as proposed in [22]. In this setting, the Bregman formulation results in an endpoint error of 0.12 (runtime: 93s.) for the *Rubberwhale* sequence and 0.11 (runtime: 45s.) for *Dimetrodon*. Brox-QDT-M achieves an endpoint error of 0.12 and a computation time of 5 seconds for each sequence. Therefore, we can again conclude that the Bregman formulation is just as accurate as other methods from the literature, but not as fast. However, one should also keep in mind that our implementation of the Bregman iteration uses a Gauß-Seidel scheme, which is much slower than the multigrid approach used in Brox-QDT-M. If both algorithms were using the same solver, then the runtimes might become comparable. For these tests we used the parameters given in Table 5.4 and Table 5.5. The multigrid solver performed 1 W-cycle, 1 pre-/postsmoothing iteration and used a Gauß-Seidel type solver with alternating line relaxation.

	Bregman L_1-L_1		Bregman L_2-L_1	
Parameter	Rubberwhale	Dimetrodon	Rubberwhale	Dimetrodon
α	0.0073	0.0020	0.0100	0.1100
γ	0	0	20.00	8.43
μ	0.35	0.41	11.25	2.30
σ	0.44	0.78	0.40	0.73
i	100	30	30	10
j	20	15	10	10
k	3	3	3	3

Table 5.4: Parameter choices for the two Bregman algorithms.

Parameter	Rubberwhale	Dimetrodon
α	0.410	0.730
θ	0.250	0.380
σ	0.376	0.605
τ	0.125	0.125
iterations	40	15
chambolle steps	15	10

Parameter	Rubberwhale	Dimetrodon
α	125.00	48.00
γ	26.60	8.43
σ	0.50	0.50

Table 5.5: **Left:** Parameter choices for TV-L_1-M. **Right:** Parameter choices for Brox-QDT-M.

5.3 Occlusion handling

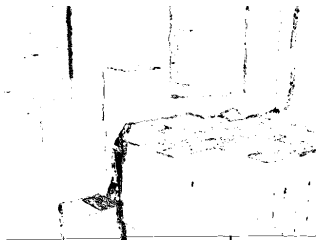
As already mentioned in the previous chapter, the quadratic penalisation of the data term in the L_2–L_1 models may require additional treatment to avoid strong outliers. We discussed one possible approach to overcome this deficit in Section 4.12. In the following we demonstrate the benefits of that approach. We consider the *Urban 2* sequence[1] and compute the flow field once without occlusion handling and another time with occlusion handling using the rotationally invariant L_2–L_1 model. The resulting displacement fields are shown in Fig. 5.4. The improvements are clearly visible in the upper left part. Without occlusion handling we get unpleasant outliers. However these outliers completely vanish if we apply the occlusion handling. These observations are also reflected in the errors. Without occlusion handling the average angular error was 3.67, whereas the average endpoint error was 0.48. The computation took about 43 seconds. Applying the occlusion handling leads to an average angular error of 3.18 and an average endpoint error of 0.41, thus yielding an improvement of almost 20% on the endpoint error. However, the computation time also rose to 242 seconds due to the computation of the multiple flow fields that are needed to determine the occluded pixels.

5.4 Summary and concluding remarks

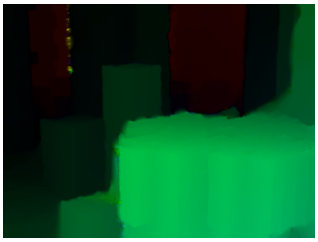
In this chapter we made a short numerical evaluation of the Bregman algorithms that we had developed in the previous chapters. We also demonstrated the benefits of an occlusion handling strategy. In general, the obtained results are relatively good and reflect the nice theoretical properties of the Bregman framework. A part of this quality can certainly also be accredited to the versatility of the Bregman toolkit that easily allows to incorporate higher order constancy assumptions in the data term as well as rotationally invariant models for the smoothness terms. Unfortunately our premonition from Remark 4.3 that the L_1-L_1 model might exhibit a slower convergence seems to be verified. In the previous tests it was by far the slowest approach. On the other hand it is interesting to note that the L_2-L_1 and L_1-L_2 model also present a certain discrepancy in their convergence speeds, although the algorithms are of a similar form since their corresponding models both have one differentiable term and one non-differentiable term. Finally, one should note that our Bregman formulations are competitive to state of the art algorithms in terms of accuracy. However, they are usually slower.

[1] also taken from the Middlebury page. See [74].

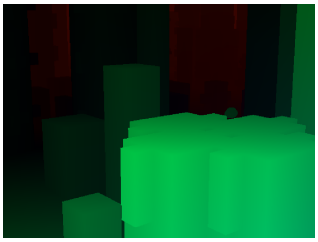
(a) Frame 10 of the *Urban 2* sequence



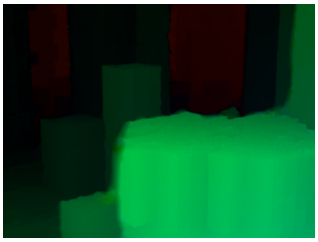
(b) Obtained occlusion indicator function for $T_{occ} = 1.30$, black pixels mark occlusions.



(c) The resulting flow field of the *Urban 2* sequence without occlusion handling. Param.: $\lambda = 0.023$, $\mu = 20.0$, $\gamma = 4.74$, $\sigma = 0.5$.



(d) Exact ground truth.



(e) The resulting flow field of the *Urban 2* sequence with occlusion handling. Param.: $\lambda = 0.2$, $\mu = 20.0$, $\gamma = 1.73$, $\sigma = 0.5$ and $T_{occ} = 1.30$.

Figure 5.4: Benefits of a proper occlusion handling. Both flow fields were computed with 10 Bregman iterations, 3 alternating minimisation steps and 10 Gauss-Seidel iterations.

6 Summary and outlook

6.1 Summary

Within the framework of this thesis we analysed the Bregman iteration of Osher et al. and proved that it was possible to minimise variational formulations of the optical flow problem by using the Bregman framework. Especially the split Bregman algorithm of Goldstein and Osher proved to be well suited for this kind of problems. Our work can basically be divided into two distinct parts.

In the first part, starting with Chapter 2, we recalled the necessary mathematical prerequisites from convex and functional analysis. All the key results and important subjects, such as the subdifferential calculus and the characteristics of shrinkage operators, were discussed and complemented with detailed proofs. In Chapter 3 we analysed the properties of various Bregman algorithms. We established a solid mathematical foundation for the upcoming tasks by discussing three of the most popular Bregman iterations: the original formulation of Osher, an alternative form of this algorithm which avoids the computation of a subgradient and the split Bregman iteration of Goldstein and Osher. One of the most important results from this chapter was the convergence of all these iterative processes. In this context it is also interesting to note that it was possible to give a concrete upper bound for the error at each iteration step. Finally, we were also able to point out a relationship between our Bregman formulation and the Lagrangian penalty method.

In the second part we turned our interest towards the optical flow problem. In Chapter 4 we developed a novel approach based upon the split Bregman algorithm and demonstrated the high flexibility of our formulation by presenting the corresponding algorithms for several different variational models. Within this context we were also able to show that the Bregman framework easily allows the integration of modern higher order data terms as well as advanced smoothness terms such as the popular total variation regulariser. In order to be able to apply the Bregman framework to problems with robust data and smoothness terms we also had to present a novel variation of the split Bregman iteration. In this new formulation we place all the terms from the energy functional into the constraining conditions. Interestingly, all our final algorithms were quite similar and relatively easy to implement. They consisted essentially of a linear system and shrinkage operations. A careful analysis even showed us that the matrix of this system is always symmetric and positive definite. This result granted us many freedoms when it came to the choice of the solver for the linear system and could be one potential starting point to further increase the efficiency of our algorithms. Furthermore, we discussed several other possible extensions and improvements. We showed that our Bregman algorithms could be incorporated into a coarse-to-fine strategy and we presented a simple and yet efficient

way to deal with strong outliers.

Finally, we verified our findings through empirical tests which proved to be very successful. Compared to other techniques in the literature, our approach performed favorably. Although the Bregman approach was usually slower, we observed that the resulting flow fields were of equal or even better quality.

6.2 Outlook

Regarding future work, research in the following directions could provide fruitful results:

The Bregman iteration itself

- The formulation presented within this thesis required a discrete setting to operate in. It is unclear whether this is really necessary. Many results presented in Chapter 2 and Chapter 3 hold for any normed vector space. Therefore, it should also be possible to use the Bregman iteration for optimisation problems in arbitrary spaces, such as the Sobolev spaces. The main problems are clearly the well definedness of the iterates. Convex functionals are only continuous in the finite dimensional case. However, without continuity we cannot guarantee the existence of a subgradient. Furthermore, one would have to investigate whether the iterates always exist, i.e. whether the minimum in each iteration step can really be attained. The calculus of variations provides results that guarantee the existence of minimisers for convex functionals, but they are often bound to rather strict conditions and may not be applicable. The authors of [5, 15] showed for example that minimisers exist if the energy functional fulfils certain growth conditions. Further results directed towards optical flow models can also be found in [48]. Osher and his colleagues proved the well definedness of the Bregman iteration for the continuous setting in [62]. However, their argumentation was specific to the ROF model for image denoising and it might be difficult to generalise their proof to other functionals.

- In [81] the authors noticed that, for certain problems, the split Bregman iteration is equivalent to an augmented Lagrangian and penalty approach. We have shown such an equivalence for a special case in Section 3.1.4. In [63] the authors showed that the linearised Bregman algorithm, another variant of the Bregman iteration that we did not mention here, is, for certain convex optimisation problems, equivalent to a gradient descent approach applied on the dual problem. Further equivalences have also been pointed out in [77]. From a mathematical point of view such equivalences are highly interesting because they often imply that one can simply transfer the knowledge about one algorithm to the other. The natural question that arises in this context is whether such equivalences still exist in a general setting and not just for specific problems. This could for example lead to a better understanding of the convergence properties of the Bregman iteration. We showed in Chapter 3 that the Bregman formulation converges for a very specific form of constraining conditions. Only very few results exist for more generic problems. By means of equivalences it might be possible to state a general convergence theory.

Better numerics
The computation of optical flow always requires the treatment of large amounts of data. Therefore, an efficient treatment of this data could provide considerable speedups to our algorithms.

- All the Bregman iterations in this thesis have one thing in common. They require solving a large and sparse linear system with a symmetric and positive definite matrix. Our tests proved that iterative approaches like the Gauß-Seidel algorithm could be used for this task. Unfortunately, this solver exhibits rather slow convergence speeds. On the other hand, multigrid methods rank among the fastest known algorithms for solving linear systems and could basically lead to a tremendous decrease in computation time if it were possible to apply them in the context of Bregman iterations. The multigrid framework has already been applied to optical flow problems in [19, 22]. The authors of these references were able to achieve speedups of more than two orders of magnitude when compared to other algorithms and reached, in certain cases, almost realtime performance. Since the multigrid algorithm is relatively complex, it might also make sense to consider other simpler methods like conjugate gradients, bicgstab and gmres. See [73, 76, 80] for a detailed analysis of these algorithms. In that context it is certainly worth analysing whether preconditioning strategies might be of any benefit. The matrix entries never change during the Bregman iteration. Therefore, one can apply the same preconditioner on the linear system at every iteration step. This observation must certainly be exploited if one wants to achieve significant performance increases. However, we have also seen in Chapter 5 that the linear system does not need to be solved with full precision. Crude approximations often work surprisingly well and may present an argument against preconditioners.

- Except for the linear system, all other operations in the split Bregman algorithm are performed pointwise and could basically be performed in parallel. Considering the increasing popularity of parallel architectures, this might be another beneficial approach for speeding up the algorithm. Formulations such as the Jacobi or Red-Black Gauß-Seidel iteration can easily be executed in parallel and would allow us to formulate a complete parallelisation of the Bregman framework.

Better models for the optical flow problem
The following ideas have not been treated yet, but they seem natural and worth investigating.

- The extension to colour images should not be too difficult and might provide slight improvements in quality. As mentioned in [20], using the different representations of the colour space and other photometric invariants could render our methods more robust against illumination changes.

- Spatiotemporal extensions using more than two frames to compute the flow might as well be beneficial to the results.

- We only considered simple quadratic or sub-quadratic smoothness and data terms within this thesis. However, during the last years more complex models have also been studied. Excellent results have been achieved by using flow and image driven smoothness terms combined with various different data terms. A detailed survey and evaluation of all these models can be found in [19, 64, 87]. Furthermore, formulations such as the complementary optical flow model [98] yield excellent results in terms of accuracy and outperform most other known approaches. Therefore, it might be worth investigating how far these model assumptions can be incorporated into the Bregman formulation.

Extensions to other correspondence problems

Motion estimation is not the only important correspondence problem in computer vision. Stereo reconstruction and medical image registration are other widely researched representatives of this type of problems. Since the Bregman algorithm proved to be useful for the computation of the optical flow, it is natural to consider whether the algorithm can also be applied to these topics. Stereo reconstruction requires the integration of the epipolar constraint [38]. This constraint limits the search space of the underlying correspondence problem to certain lines (epipolar lines) and the number of unknowns is reduced to one. However, the structure of the equations gets significantly more complex and is likely to cause many difficulties if one wanted to apply a Bregman formulation. Similar problems are also to be expected in the context of medical image registration, where one seeks the deformation field that allows to compare/combine information from different image acquisition methods [20]. On the one hand one may have to consider models that respect certain physical constraints and on the other hand one must also be able to handle images that originate from completely different sources. This certainly presents a non-trivial task, both in terms of modeling and numerics.

Bibliography

All cited webpages were available in August 2010.

[1] E. Albrecht. Funktionalanalysis 1 und 2. Vorlesungsskript Wintersemester 2008/2009 und Sommersemester 2009 Universität des Saarlandes, 2008, 2009. `http://www.math.uni-sb.de/ag/albrecht/ag-albrecht.html`.

[2] H. W. Alt. *Lineare Funktionalanalysis*. Springer-Lehrbuch Masterclass. Springer, 5th edition, 2006.

[3] W. Alt. *Numerische Verfahren der konvexen Optimierung: Eine anwendungsorientierte Einführung*. Vieweg+Teubner, 1st edition, 2004.

[4] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2(3):283–310, 1989.

[5] G. Aubert and P. Kornprobst. *Mathematical problems in image processing, partial differential equations and the calculus of variations*, volume 147 of *Applied Mathematical Sciences*. Springer, 2nd edition, 2006.

[6] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *Proceedings of the 2007 IEEE International Conference on Computer Vision*, pages 1–8, Rio de Janeiro, Brazil, October 2007. IEEE Computer Society Press.

[7] J. Barron, D. J. Fleet, and S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[8] H. H. Bauschke and J. M. Borwein. Legendre functions and the method of random Bregman projections. *Journal of Convex Analysis*, 4(1):27–67, 1997.

[9] R. Ben-Ari and N. Sochen. Variational stereo vision with sharp discontinuities and occlusion handling. In *Proc. 2007 IEEE International Conference on Computer Vision*, Rio de Janeiro, Brazil, 2007. IEEE Computer Society Press.

[10] M. Black and P. Anandan. The robust estimation of multiple motions: parametric and piecewise smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, 1996.

[11] M. J. Black and P. Anandan. Robust dynamic motion estimation over time. In *Proceedings of the 1991 IEEE Computer Society Conference on Computer Vision*

and Pattern Recognition, pages 292–302, Maui, Hawaii, June 1991. IEEE Computer Society Press.

[12] J. M. Borwein and A. S. Lewis. *Convex analysis and nonlinear optimization: Theory and examples*. CMS Books in Mathematics. Springer, 1st edition, 2000.

[13] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 1st edition, 2004. Availabe from: `http://www.stanford.edu/~boyd/cvxbook/`.

[14] L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217, 1967.

[15] D. Breit. Grundlagen der variationsrechnung I & II. Vorlesungsskript Wintersemester 2009/2010 und Sommersemester 2010 Universität des Saarlandes, 2009, 2010. `http://www.math.uni-sb.de/ag/fuchs/GdV/gdv.html`.

[16] M. Breuß. Numerical algorithms for visual computing III. Lecture summer semester 2009 Saarland University, 2009. `http://www.mia.uni-saarland.de/breuss/index.shtml`.

[17] M. Brokate. Konvexe Analysis. Vorlesungsskript Sommersemester 2008 und 2009 TU München, 2008, 2009. `http://www-m6.ma.tum.de/~brokate/`.

[18] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In T. Pajdla and J. Matas, editors, *Computer Vision – ECCV 2004, Part IV*, volume 3024 of *Lecture Notes in Computer Science*, pages 25–36. Springer, 2004.

[19] A. Bruhn. *Variationelle Optische Flussberechnung Präzise Modellierung und effiziente Numerik*. PhD thesis, Saarland University, 2006.

[20] A. Bruhn. Correspondence problems in computer vision. Lecture summer semester 2009 Saarland University, 2009. `http://www.mia.uni-saarland.de/bruhn/index.shtml`.

[21] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr. Variational optic flow computation in real-time. Technical Report 89, Department of Mathematics, Saarland University, Saarbrücken, 2003.

[22] A. Bruhn, J. Weickert, T. Kohlberger, and C. Schnörr. A multigrid platform for real-time motion computation with discontinuity-preserving variational methods. *International Journal of Computer Vision*, 70(3):257–277, 2006.

[23] C. Brune, A. Sawatzky, and M. Burger. Primal and dual Bregman methods with application to optical nanoscopy. Technical report, Westfälische Wilhelms-Universität Münster, Institut für numerische und angewandte Mathematik, 2010. Submitted to International Journal of Computer Vision.

[24] M. Burger, G. Gilboa, S. Osher, and J. Xu. Nonlinear inverse scale space methods. *Communications in Mathematical Sciences*, 4(1):175–208, 2006.

[25] M. Burger and S. Osher. Convergence rates of convex variational regularization. *Inverse Problems*, 20(5):1411–1420, 2004.

[26] M. Burger, S. Osher, J. Xu, and G. Gilboa. Nonlinear inverse scale space methods for image restoration. In N. Paragios, O. D. Faugeras, T. Chan, and C. Schnörr, editors, *Variational, Geometric, and Level Set Methods in Computer Vision (VLSM). Third international workshop*, volume 3752 of *Lecture Notes in Computer Science*, pages 25–36, Beijing, China, October 2005. Springer.

[27] M. Burger, E. Resmerita, and L. He. Error estimation for Bregman iterations and inverse scale space methods. *Computing*, 81(2–3):109–135, 2007.

[28] J.-F. Cai, S. Osher, and Z. Shen. Linearized Bregman iterations for compressed sensing. *Mathematics of Computation*, 78(267):1515–1536, 2009.

[29] J.-F. Cai, S. Osher, and Z. Shen. Split Bregman methods and frame based image restoration. *Multiscale Modeling & Simulation*, 8(2):337–369, 2009.

[30] A. Chambolle. An algorithm for total variation minimization and applications. *Journal of Mathematical Imaging and Vision*, 20(1–2):89–97, 2004.

[31] T. F. Chan and P. Mulet. On the convergence of the lagged diffusivity fixed point method in total variation image restoration. *SIAM Journal on Numerical Analysis*, 36(2):354–367, 1999.

[32] P. Chen, Y. Chen, and M. Rao. Metrics defined by Bregman divergences. *Communications in Mathematical Sciences*, 6(4):915–926, 2008.

[33] P. Chen, Y. Chen, and M. Rao. Metrics defined by Bregman divergences: Part 2. *Communications in Mathematical Sciences*, 6(4):927–948, 2008.

[34] S. Cochran and G. Medioni. 3-D surface description from binocular stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):981–994, 1992.

[35] I. Cohen. Nonlinear variational method for optical flow computation. In K. A. Høgda, B. Braathen, and K. Heia, editors, *Proceedings of the 8th Scandinavian Conference on Image Analysis*, volume 1, pages 523–530, Tromsø, Norway, May 1993. Nobim.

[36] R. Deriche, P. Kornprobst, and G. Aubert. Optical flow estimation while preserving its discontinuities: a variational approach. In S. Z. Li, D. P. Mital, E. K. Teoh, and H. Wang, editors, *Recent Developments in Computer Vision, Second Asian Conference on Computer Vision, ACCV '95*, volume 1035 of *Lecture Notes in Computer Science*, pages 69–80, Singapore, December 1996. Springer.

[37] I. Ekeland and R. Téman. *Convex analysis and variational problems*. Number 28 in Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1st edition, 1999.

[38] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. Artificial Intelligence. MIT Press, 1st edition, 1993.

[39] S. Fučik, A. Kratochvil, and J. Nečas. Kačanov–Galerkin method. *Commentationes Mathematicae Universitatis Carolinae*, 14(4):651–659, 1973.

[40] C. Geiger and C. Kanzow. *Theorie und Numerik restringierter Optimierungsaufgaben*. Springer-Lehrbuch Masterclass. Springer, 1st edition, 2002.

[41] P. Getreuer. Notes on Bregman iteration. Online, 2009. `http://www.math.ucla.edu/~getreuer/`.

[42] T. Goldstein, X. Bresson, and S. Osher. Global minimization of Markov random fields with applications to optical flow. UCLA CAM Report 09-77, University of California, Los Angeles, 2009.

[43] T. Goldstein and S. Osher. The split Bregman method for l_1 regularized problems. *SIAM Journal on Imaging Sciences*, 2(2):323–343, 2009.

[44] A. Göpfert, T. Riedrich, and C. Tammer. *Angewandte Funktionalanalysis: Motivationen und Methoden für Mathematiker und Wirtschaftswissenschaftler*. Vieweg+Teubner, 1st edition, 2009.

[45] W. Hackbusch. *Multigrid Methods and Applications*. Springer series in computational mathematics. Springer, 1st edition, 2009.

[46] E. T. Hale, W. Yin, and Y. Zhang. A fixed-point continuation method for l_1 regularized minimization with applications to compressed sensing. CAAM Technical Report TR07-07, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 2007.

[47] L. He, T.-C. Chang, S. Osher, T. Fang, and P. Speier. MR image reconstruction by using the iterative refinement method and nonlinear inverse scale space methods. UCLA CAM Report 06-35, University of California, Los Angeles, 2006.

[48] W. Hinterberger, O. Scherzer, C. Schnörr, and J. Weickert. Analysis of optical flow models in the framework of calculus of variations. *Numerical Functional Analysis and Optimization*, 23(1 & 2):69–89, 2002.

[49] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex analysis and minimization algorithms I: Fundamentals*. A series of comprehensive studies in mathematics. Springer, 2nd edition, 1996.

[50] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1–3):185–203, 1981.

[51] V. John. Mathematische Optimierung. Vorlesungsskript Sommersemester 2009 Universität des Saarlandes, 2009. `http://www.wias-berlin.de/people/john/`.

[52] D. Jungnickel. *Optimierungsmethoden*. Springer Lehrbuch. Springer, 2nd edition, 2008.

[53] Y. Kameda, A. Imiya, and N. Ohnishi. A convergence proof for the Horn-Schunck optical-flow computation scheme using neighborhood decomposition. In V. E. Brimkov, R. P. Barneva, and H. A. Hauptman, editors, *Proceedings of the 12th international conference on Combinatorial image analysis*, volume 4958 of *Lecture Notes in Computer Science*, pages 262–273, Buffalo, USA, April 2008. Springer.

[54] J. Kačur, J. Nečas, J. Polák, and J. Souček. Convergence of a method for solving the magnetostatic field in nonlinear media. *Aplikace Matematiky*, 13:456–465, 1968.

[55] R. Klette, K. Schlüns, and A. Koschan. *Computer Vision: Three-Dimensional Data from Images*. Springer, 1st edition, 1998.

[56] A. Mansouri, A. Mitiche, and J. Konrad. Selective image diffusion: application to disparity estimation. In *Proc. 1998 IEEE International Conference on Image Processing*, volume 3, pages 284–288, Chicago, IL, 1998.

[57] A. Mitiche and A.-R. Mansouri. On convergence of the Horn and Schunck optical-flow estimation method. *IEEE Transactions on Image Processing*, 13(6):848–852, 2004.

[58] E. Ménim and P. Pérez. Hierarchical estimation and segmentation of dense motion fields. *International Journal of Computer Vision*, 46(2):129–155, 2002.

[59] P. Mrázek, J. Weickert, and G. Steidl. Correspondences between wavelet shrinkage and nonlinear diffusion. In L. Griffin and M. Lillholm, editors, *Scale-Space 2003*, volume 2695 of *Lecture notes in computer science*, pages 101–116. Springer, 2003.

[60] H. Nagel. Image sequence server. Online. `http://i21www.ira.uka.de/image_sequences/`.

[61] T. Nir, A. Bruckstein, and R. Kimmel. Over-parameterized variational optical flow. *International Journal of Computer Vision*, 76(2):205–216, 2008.

[62] S. Osher, M. Burger, D. Goldfarb, J. Xu, and W. Yin. An iterative regularization method for total variation-based image restoration. *Multiscale Modeling and Simulation*, 4(2):460–489, 2005.

[63] S. Osher, Y. Mao, B. Dong, and W. Yin. Fast linearized Bregman iteration for compressive sensing and sparse denoising. *Communications in Mathematical Sciences*, 8(1):93–111, 2010.

[64] N. Papenberg, A. Bruhn, T. Brox, S. Didas, and J. Weickert. Highly accurate optic flow computation with theoretically justified warping. *International Journal of Computer Vision*, 67(2):141–158, 2006.

[65] R. Plato. *Numerische Mathematik kompakt*. Grundlagenwissen für Studium und Praxis. Vieweg+Teubner, 4th edition, 2010.

[66] T. Pock. *Fast total variation for computer vision*. PhD thesis, Graz University of Technology, 2008.

[67] M. Proesmans, L. V. Gool, E. Pauwels, and A. Oosterlinck. Determination of optical flow and its discontinuities using non-linear diffusion. In J.-O. Eklundh, editor, *Computer Vision – ECCV '94*, volume 801 of *Lecture Notes in Computer Science*, pages 295–304. Springer, 1994.

[68] A. W. Roberts and D. E. Varberg. *Convex functions*. Pure & Applied Mathematics. Academic Press Inc., 1st edition, 1973.

[69] R. T. Rockafellar. *Convex analysis*. Princeton Landmarks in Mathematics and Physics. Princeton University Press, 10th edition, 1997.

[70] R. T. Rockafellar and J.-B. Wets. *Variational Analysis*, volume 317 of *A series of comprehensive studies in mathematics*. Springer, 1st edition, 1998.

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

[72] W. Rudin. *Functional Analysis*. International series in pure and applied mathematics. McGraw-Hill, 2nd edition, 1991.

[73] Y. Saad. *Iterative methods for sparse linear systems*. Society for industrial mathematics, 2nd edition, 2000. Available from `http://www-users.cs.umn.edu/~saad/books.html`.

[74] D. Scharstein and R. Szeliski. The middlebury computer vision pages. Online. `http://vision.middlebury.edu/`.

[75] C. Schnörr. Determining optical flow for irregular domains by minimizing quadratic functionals of a certain class. *International Journal of Computer Vision*, 6(1):25–38, 1991.

[76] H. R. Schwarz and N. Köckler. *Numerische Mathematik*. Teubner, 6th edition, 2006.

[77] S. Setzer. Split Bregman algorithm, Douglas-Rachford splitting and frame shrinkage. In X.-C. Tai, K. Mørken, M. Lysaker, and K.-A. Lie, editors, *Scale Space and Variational Methods in Computer Vision*, volume 5567 of *Lecture Notes in Computer Science*, pages 464–476, Voss, Norway, June 2009. Springer.

[78] N. Slesareva. Dense disparity map estimation – A novel approach based on high accuracy optic flow computation. Master's thesis, Saarland University, 2005.

[79] G. Steidl, J. Weickert, T. Brox, P. Mrázek, and M. Welk. On the equivalence of soft wavelet shrinkage, total variation diffusion, total variation regularization and sides. *SIAM Journal on Numerical Analysis*, 42(2):686–713, 2004.

[80] J. Stoer and R. Bulirsch. *Numerische Mathematik 2*. Springer Lehrbuch. Springer, 5th edition, 2005.

[81] X.-C. Tai and C. Wu. Augmented Lagrangian method, dual methods and split Bregman iteration for ROF model. In X.-C. Tai, K. Mørken, M. Lysaker, and K.-A. Lie, editors, *Scale Space and Variational Methods in Computer Vision*, volume 5567 of *Lecture Notes in Computer Science*, pages 502–513, Voss, Norway, June 2009. Springer.

[82] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, Englewood Cliffs, 1st edition, 1998.

[83] M. Ulbrich. Funktionalanalysis und Anwendungen. Vorlesungsskript Wintersemester 2004/2005 Universität Hamburg, 2005. `http://www.math.uni-hamburg.de/home/ulbrich/fa/`.

[84] Y. Wang, W. Yin, and Y. Zhang. A fast algorithm for image deblurring with total variation regularization. CAAM Technical Report TR07-10, Department of Computational and Applied Mathematics, Rice University, Houston, Texas, 2007.

[85] A. Wedel, D. Cremers, T. Pock, and H. Bischof. Structure- and motion-adaptive regularization for high accuracy optic flow. In *Proc. 2009 IEEE International Conference on Computer Vision*, Kyoto, Japan, 2009. IEEE Computer Society Press.

[86] A. Wedel, T. Pock, C. Zach, H. Bischof, and D. Cremers. An improved algorithm for TV–L^1 optical flow computation. In D. Cremers, B. Rosenhahn, A. L. Yuille, and F. R. Schmidt, editors, *Statistical and Geometrical Approaches to Visual Motion Analysis*, volume 5604 of *Lecture Notes in Computer Science*, pages 23–45. Springer, 2008.

[87] J. Weickert, A. Bruhn, T. Brox, and N. Papenberg. A survey on variational optic flow methods for small displacements. Technical Report 152, Department of Mathematics, Saarland University, Saarbrücken, 2005.

[88] J. Weickert, A. Bruhn, N. Papenberg, and T. Brox. Variational optic flow computation: From continuous models to algorithms. In L. Alvarez, editor, *IWCVIA '03: International Workshop on Computer Vision and Image Analysis*, volume 0026, pages 1–6, Spain, 2004. Cuadernos del Instituto Universitario de Ciencias y Tecnologias Ciberneticas, University of Las Palmas de Gran Canaria.

[89] J. Weickert and C. Schnörr. A theoretical framework for convex regularizers in pde-based computation of image motion. *International Journal of Computer Vision*, 45(3):245–264, 2001.

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

[91] D. Werner. *Funktionalanalysis*. Springer-Lehrbuch. Springer, 6th edition, 2007.

[92] J. Xu and S. Osher. Iterative regularization and nonlinear inverse scale space applied to wavelet-based denoising. *IEEE Transactions on Image Processing*, 16(2):534–544, 2006.

[93] L. Xu, J. Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. In *Proc. 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, USA, 2010. IEEE Computer Society Press.

[94] W. Yin, S. Osher, D. Goldfarb, and J. Darbon. Bregman iterative algorithms for l1-minimization with applications to compressed sensing. *SIAM Journal on Imaging Sciences*, 1(1):143–168, 2007.

[95] C. Zach, T. Pock, and B. Horst. A duality based approach for realtime TV–L1 optical flow. In F. A. Hamprecht, C. Schnörr, and B. Jähne, editors, *Proceedings of the 29th DAGM Symposium on Pattern Recognition*, volume 4713 of *Lecture Notes in Computer Science*, pages 214–223, Heidelberg, Germany, September 2007. Springer.

[96] X. Zhang, M. Burger, X. Bresson, and S. Osher. Bregmanized nonlocal regularization for deconvolution and sparse reconstruction. UCLA CAM Report 09-03, University of California, Los Angeles, 2009.

[97] H. Zimmer, M. Breuß, J. Weickert, and H.-P. Seidel. Hyperbolic numerics for variational approaches to correspondence problems. In X.-C. T. et al., editor, *Scale-Space and Variational Methods in Computer Vision*, volume 5567 of *Lecture Notes in Computer Science*, pages 636–647. Springer, 2009.

[98] H. Zimmer, A. Bruhn, J. Weickert, L. Valgaerts, A. Salgado, B. Rosenhahn, and H.-P. Seidel. Complementary optic flow. In D. Cremers, Y. Boykov, A. Blake, and F. R. Schmidt, editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, volume 5681 of *Lecture notes in computer science*, pages 207–220. Springer, 2009.