Acta Numerica (2016), pp. 161–319
doi:10.1017/S096249291600009X
© Cambridge University Press, 2016
Printed in the United Kingdom

An introduction to continuous optimization for imaging

Antonin Chambolle
CMAP, Ecole Polytechnique, CNRS, France
E-mail: antonin.chambolle@cmap.polytechnique.fr

Thomas Pock
ICG, Graz University of Technology, AIT, Austria
E-mail: pock@icg.tugraz.at

A large number of imaging problems reduce to the optimization of a cost function, with typical structural properties. The aim of this paper is to describe the state of the art in continuous optimization methods for such problems, and present the most successful approaches and their interconnections. We place particular emphasis on optimal first-order schemes that can deal with typical non-smooth and large-scale objective functions used in imaging problems. We illustrate and compare the different algorithms using classical non-smooth problems in imaging, such as denoising and deblurring. Moreover, we present applications of the algorithms to more advanced problems, such as magnetic resonance imaging, multilabel image segmentation, optical flow estimation, stereo matching, and classification.

CONTENTS

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

1. Introduction

The purpose of this paper is to describe, and illustrate with numerical examples, the fundamentals of a branch of continuous optimization dedicated to problems in imaging science, in particular image reconstruction, inverse problems in imaging, and some simple classification tasks. Many of these problems can be modelled by means of an 'energy', 'cost' or 'objective' which represents how 'good' (or bad!) a solution is, and must be minimized.

These problems often share a few characteristic features. One is their size, which can be very large (typically involving at most around a billion variables, for problems such as three-dimensional image reconstruction, dense stereo matching, or video processing) but usually not 'huge' like some recent problems in learning or statistics. Another is the fact that for many problems, the data are structured in a two- or three-dimensional grid and interact locally. A final, frequent and fundamental feature is that many useful problems involve non-smooth (usually convex) terms, for reasons that are now well understood and concern the concepts of sparsity (DeVore 1998, Candès, Romberg and Tao 2006*b*, Donoho 2006, Aharon, Elad and Bruckstein 2006) and robustness (Ben-Tal and Nemirovski 1998).

These features have strongly influenced the type of numerical algorithms used and further developed to solve these problems. Due to their size and lack of smoothness, higher-order methods such as Newton's method, or methods relying on precise line-search techniques, are usually ruled out, although some authors have suggested and successfully implemented quasi-Newton methods for non-smooth problems of the kind considered here (Ito and Kunisch 1990, Chan, Golub and Mulet 1999).

Hence these problems will usually be tackled with first-order descent methods, which are essentially extensions and variants of a plain gradient descent, appropriately adapted to deal with the lack of smoothness of the objective function. To tackle non-smoothness, one can either rely on controlled smoothing of the problem (Nesterov 2005, Becker, Bobin and Candès 2011) and revert to smooth optimization techniques, or 'split' the problem into smaller subproblems which can be exactly (or almost) solved, and combine these resolutions in a way that ensures that the initial problem is eventually solved. This last idea is now commonly referred to as 'proximal splitting' and, although it relies on ideas from as far back as the 1950s or 1970s (Douglas and Rachford 1956, Glowinski and Marroco 1975), it has been a very active topic in the past ten years in image and signal processing, as well as in statistical learning (Combettes and Pesquet 2011, Parikh and Boyd 2014).

Hence, we will focus mainly on proximal splitting (descent) methods, and primarily for convex problems (or extensions, such as finding zeros of maximal-monotone operators). We will introduce several important

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

problems in imaging and describe in detail simple first-order techniques to solve these problems practically, explaining how 'best' to implement these methods, and in particular, when available, how to use acceleration tricks and techniques to improve the convergence rates (which are generally very poor for such methods). This point of view is very similar to the approach in a recent tutorial of Burger, Sawatzky and Steidl (2014), though we will describe a larger class of problems and establish connections between the most commonly used first-order methods in this field.

Finally, we should mention that for many imaging problems, the grid structure of the data is well suited for massively parallel implementations on GPUs, and hence it is beneficial to develop algorithms that preserve this property.

The organization of this paper is as follows. We will first describe typical (simple) problems in imaging and explain how they can be reduced to the minimization of relatively simple functions, usually convex. Then, after a short introduction to the basic concepts of convexity in Section 3, we will describe in Sections 4 and 5 the classes of algorithms that are currently used to tackle these problems, illustrating each algorithm with applications to the problems introduced earlier. Each time, we will discuss the basic methods, convergence results and expected rates, and, when available, acceleration tricks which can sometimes turn a slow and inefficient method into a useful practical tool. We will focus mainly on two families of methods (whose usefulness depends on the structure of the problem): first-order descent methods and saddle-point methods. Both can be seen as either variants or extensions of the 'proximal-point algorithm' (Martinet 1970), and are essentially based on iterations of a 1-Lipschitz operator; therefore, in Appendix A we will very briefly recall the general theory for such iterative techniques. It does not apply to accelerated variants which are not usually contractive (or not known to be contractive), but rates of convergence can be estimated; see Appendices B and C.

In a final theoretical section (Section 6) we will briefly introduce some extensions of these techniques to non-convex problems.

Then, in Section 7, we will review a series of practical problems (*e.g.*, first- and higher-order regularization of inverse problems, feature selection and dictionary learning, segmentation, basic inpainting, optical flow), each time explaining which methods can be used (and giving the implementations in detail), and how methods can be tuned to each problem. Of course, we do not claim that we will always give the 'optimal' method to solve a problem, and we will try to refer to the relevant literature where a more thorough study can be found.

Our review of first-order algorithms for imaging problems is partly inspired by our own work and that of many colleagues, but also by important textbooks in optimization (Polyak 1987, Bertsekas 2015, Ben-Tal and

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Nemirovski 2001, Nesterov 2004, Boyd and Vandenberghe 2004, Nocedal and Wright 2006, Bauschke and Combettes 2011). However, we have tried to keep the level of detail as simple as possible, so that most should be accessible to readers with very little knowledge of optimization theory. Naturally we refer the interested reader to these references for a deeper understanding of modern optimization.

Finally we should mention that we will overlook quite a few important problems and methods in imaging. First, we will not discuss combinatorial optimization techniques for regularization/segmentation, as we fear that this would require us to almost double the size of the paper. Such methods, based on graph cuts or network flows, are very efficient and have been extensively developed by the computer vision community to tackle most of the problems we address here with continuous optimization. As an example, the paper of Boykov, Veksler and Zabih (2001), which shows how to minimize the 'Potts' model (7.25) using graph-cuts, attains almost 6000 citations in *Google Scholar*, while the maximal flow algorithm of Boykov and Kolmogorov (2004) is cited more than 3500 times. We believe the two approaches complement one another nicely: they essentially tackle the same sort of problems, with similar structures, but from the perspective of implementation they are quite different. In particular, Hochbaum (2001) presents an approach to solve exactly a particular case of Problem 2.6 in polynomial time; see also Darbon and Sigelle (2006*a*, 2006*b*) (the variant in Chambolle and Darbon 2012 might be more accessible for the reader unfamiliar with combinatorial optimization). In general, graph-based methods are harder to parallelize, and can approximate fewer general energies than methods based on continuous optimization. However, they are almost always more efficient than non-parallel iterative continuous implementations for the same problem.

We will also ignore a few important issues and methods in image processing: we will not discuss many of the 'non-local' methods, which achieve state of the art for denoising (Dabov, Foi, Katkovnik and Egiazarian 2007, Buades, Coll and Morel 2005, Buades, Coll and Morel 2011). Although these approaches were not introduced as 'variational' methods, it is now known that they are closely related to methods based on structured sparsity (Danielyan, Katkovnik and Egiazarian 2012) or (patch-based) Gaussian mixture models (Mallat and Yu 2010, Yu, Sapiro and Mallat 2012, Lebrun, Buades and Morel 2013) and can be given a 'variational' form (Gilboa, Darbon, Osher and Chan 2006, Kindermann, Osher and Jones 2005, Peyré, Bougleux and Cohen 2008, Arias, Facciolo, Caselles and Sapiro 2011). The numerical algorithms to tackle these problems still need a lot of specific tuning to achieve good performance. We will address related issues in Section 7.12 (on 'Lasso'-type problems) and present alternatives to non-local denoising.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Moreover, we will not mention the recent developments in computer vision and learning based on *convolutional neural networks*, or CNNs (LeCun *et al.* 1989, Krizhevsky, Sutskever and Hinton 2012), which usually achieve the best results in classification and image understanding. These models (also highly non-local) are quite different from those introduced here, although there is a strong connection with dictionary learning techniques (which could be seen as a basic 'first step' of CNN learning). Due to the complexity of the models, the optimization techniques for CNNs are very specific and usually rely on stochastic gradient descent schemes for smoothed problems, or stochastic subgradient descent (Krizhevsky *et al.* 2012, LeCun, Bottou, Orr and Muller 1998*b*). The second author of this paper has recently proposed a framework which in some sense bridges the gap between descent methods or PDE approaches and CNN-based learning (Chen, Ranftl and Pock 2014*b*).

More generally, we will largely ignore recent developments in stochastic first-order methods in optimization, which have been driven by big data applications and the need to optimize huge problems with often billions of variables (in learning and statistics, hence also with obvious applications to image analysis and classification). We will try to provide appropriate references when efficient stochastic variants of the methods described have recently been developed.

We now describe, in the next section, the key exemplary optimization problems which we are going to tackle throughout this paper.

2. Typical optimization problems in imaging

First let us give the reader a taste of typical optimization problems that arise from classical models in image processing, computer vision and machine learning. Another of our goals is to give a short overview of typical applications of variational models in imaging; more specific models will then be described in Section 7. Among the most important features in images are edges and texture. Hence, an important property of models in image processing is the ability to preserve sharp discontinuities in their solutions in order to keep precise identification of image edges. Another goal of most models is robustness (Ben-Tal and Nemirovski 1998, Ben-Tal, El Ghaoui and Nemirovski 2009), that is, the solution of a model should be stable in the presence of noise or outliers. In practice this implies that successful models should be non-smooth and hence non-differentiable. Indeed, a successful approach to these issues is known to be realized by the minimization of robust error functions based on norm functions. Classical optimization algorithms from non-linear optimization, such as gradient methods, Newton or quasi-Newton methods, cannot be used 'out of the box' since these algorithms require a certain smoothness of the objective function or cannot be

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

applied to large-scale problems – hence the need for specialized algorithms that can exploit the structure of the problems and lead efficiently to good solutions.

2.1. Sparse representations

An important discovery in recent years (Candès *et al.* 2006*b*, Donoho 2006, Aharon *et al.* 2006) is the observation that many real-world signals can be modelled via sparse representation in a suitable basis or 'dictionary'. This property can be used to reconstruct a signal from far fewer measurements than required by the Shannon–Nyquist sampling theorem, for example, which states that the sampling frequency should be at least twice as high as the highest frequency in the signal. Furthermore, a sparse representation of a signal is desirable since it implies a certain robustness in the presence of noise. Given an input signal $b \in \mathbb{R}^m$, a sparse representation in the dictionary $A = (a_{i,j})_{i,j} \in \mathbb{R}^{m \times n}$ of n column vectors $(a_{i,j})_{i=1}^m$ can be found by solving the following optimization problem (Mallat and Zhang 1993, Chen, Donoho and Saunders 1998):

$$\begin{aligned} &\min_x f(x) \\ &\text{such that } Ax = b, \end{aligned} \tag{2.1}$$

where $x \in \mathbb{R}^n$ is the unknown coefficient vector. This model is usually known by the name *basis pursuit* (Chen and Donoho 1994). Since each column of A can be interpreted as a basis atom, the equality constraint $Ax = b$ describes the fact that the signal b should be represented as a sparse linear combination of those atoms. The function $f(x)$ is a *sparsity-inducing function*, such as $f(x) = \|x\|_1 := \sum_i |x_i|$ in the most simple case.

If some further prior knowledge concerning a relevant group structure is available, one can encode such information in the sparsity-inducing function. This idea is known as *group sparsity*, and is widely used in data analysis. It consists in using $\ell_{1,p}$-norms, with $p = 2$ or $p = \infty$. The p-norm is taken within the groups and the 1-norm is taken between the groups. This forces the solution to have only a few active groups, but within the active groups the coefficients can be dense.

For problems such as matrix factorization (Paatero and Tapper 1994, Lee and Seung 1999) or robust principal component analysis (Candès, Li, Ma and Wright 2011), where x is tensor-valued, the sparsity-inducing norm could also be a function promoting the sparsity of the singular values of x and hence forcing x to be of low rank. A popular choice to achieve this goal is the 1-Schatten norm (or nuclear norm) $\|\cdot\|_{\mathcal{S}_1}$, which is given by the 1-norm of the singular values of x, and is polar to the spectral/operator norm $\|\cdot\|_{\mathcal{S}_\infty}$.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

A more general formulation that also allows for noise in the observed signal b is given by the following optimization problem, popularized by the name 'Lasso', *least absolute shrinkage and selection operator* (Tibshirani 1996):

$$\min_x \|x\|_1 + \frac{\lambda}{2}\|Ax - b\|_2^2, \tag{2.2}$$

where $\lambda > 0$ is a parameter that can be adapted to the noise level of b. The parameter λ can also be interpreted as a Lagrange multiplier for the constraint $\frac{1}{2}\|Ax - b\|_2^2 \leq \sigma^2$, where σ is an estimate of the noise level. This shows the close connection between (2.1) and (2.2).

The Lasso approach can also be interpreted as a model that tries to *synthesize* the given signal b using only a small number of basis atoms. A closely related problem is obtained by moving the linear operator A from the data-fitting term to the regularization term, that is,

$$\min_x \|Bx\|_1 + \frac{\lambda}{2}\|x - b\|_2^2, \tag{2.3}$$

where B is again a linear operator. If A is invertible and $B = A^{-1}$, a simple change of variables shows that the two problems are equivalent. However, the more interesting cases are for non-invertible B, and the two problems can have very different properties. Here, the linear operator B can be interpreted as an operator *analysing* the signal, and hence the model is known as the *co-sparse analysis model* (Nam, Davies, Elad and Gribonval 2013). The basic idea behind this approach is that the scalar product of the signal with a given family of filters should vanish most of the time. The most influential model in imaging utilizing such sparse analysis regularizers is the *total variation* regularizer.

Here, we recall the 'ROF' (Rudin, Osher and Fatemi 1992, Chambolle and Lions 1997) model for total variation based image denoising. We consider a scalar-valued digital image $u \in \mathbb{R}^{m\times n}$ of size $m \times n$ pixels.[1] A simple and standard approach for defining the (discrete) total variation is to use a finite difference scheme acting on the image pixels. We introduce a discrete gradient operator $\mathrm{D} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n\times 2}$, which is defined by

$$\begin{aligned}(\mathrm{D}u)_{i,j,1} &= \begin{cases} u_{i+1,j} - u_{i,j} & \text{if } 1 \leq i < m, \\ 0 & \text{else,} \end{cases} \\ (\mathrm{D}u)_{i,j,2} &= \begin{cases} u_{i,j+1} - u_{i,j} & \text{if } 1 \leq j < n, \\ 0 & \text{else.} \end{cases}\end{aligned} \tag{2.4}$$

[1] Of course, what follows is also valid for images/signals defined on a one- or three-dimensional domain.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

We will also frequently need the operator norm $\|\mathrm{D}\|$, which is estimated as

$$\|\mathrm{D}\| \leq \sqrt{8} \tag{2.5}$$

(see Chambolle 2004*b*). The discrete ROF model is then defined by

$$\min_u \lambda \|\mathrm{D}u\|_{p,1} + \frac{1}{2}\|u - u^\diamond\|_2^2, \tag{2.6}$$

where $u^\diamond \in \mathbb{R}^{m\times n}$ is the given noisy image, and the discrete total variation is defined by

$$\|\mathrm{D}u\|_{p,1} = \sum_{i=1,j=1}^{m,n} |(\mathrm{D}u)_{i,j}|_p = \sum_{i=1,j=1}^{m,n} \left((\mathrm{D}u)_{i,j,1}^p + (\mathrm{D}u)_{i,j,2}^p\right)^{1/p},$$

that is, the ℓ_1-norm of the p-norm of the pixelwise image gradients.[2] The parameter p can be used, for example, to realize anisotropic ($p = 1$) or isotropic ($p = 2$) total variation. Some properties of the continuous model, such as the co-area formula, carry over to the discrete model only if $p = 1$, but the isotropic total variation is often preferred in practice since it does not exhibit a grid bias.

From a sparsity point of view, the idea of the total variation denoising model is that the ℓ_1-norm induces sparsity in the gradients of the image, hence it favours piecewise constant images with sparse edges. On the other hand, this property – also known as the *staircasing effect* – might be considered a drawback for some applications. Some workarounds for this issue will be suggested in Example 4.7 and Section 7.2. The isotropic case ($p = 2$) can also be interpreted as a very simple form of group sparsity, grouping together the image derivatives in each spatial dimension.

In many practical problems it is necessary to incorporate an additional linear operator in the data-fitting term. Such a model is usually of the form

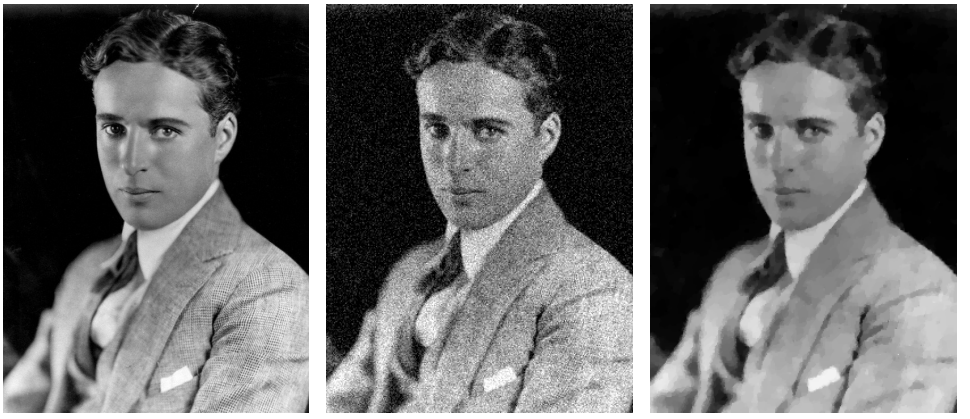
$$\min_u \lambda \|\mathrm{D}u\|_{p,1} + \frac{1}{2}\|Au - u^\diamond\|_2^2, \tag{2.7}$$

where $A : \mathbb{R}^{m\times n} \to \mathbb{R}^{k\times l}$ is a linear operator, $u^\diamond \in \mathbb{R}^{k\times l}$ is the given data, and k, l will depend on the particular application. Examples include image deblurring, where A models the blur kernel, and magnetic resonance imaging (MRI), where the linear operator is usually a combination of a Fourier transform and the coil sensitivities; see Section 7.4 for details.

The quadratic data-fitting term of the ROF model is specialized for zero-mean Gaussian noise. In order to apply the model to other types of noise, different data-fitting terms have been proposed. When the noise is impulsive or contains gross outliers, a simple yet efficient modification is to replace

[2] Taking only right differences is of course arbitrary, and may lead to anisotropy issues. However, this is rarely important for applications (Chambolle, Levine and Lucier 2011).

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) original image (b) noisy image (c) denoised image

Figure 2.1. Total variation based image denoising. (a) Original input image, and (b) noisy image containing additive Gaussian noise with standard deviation $\sigma = 0.1$. (c) Denoised image obtained by minimizing the ROF model using $\lambda = 0.1$.

the quadratic data-fitting term with an ℓ_1-data term. The resulting model, called the TV-ℓ_1 model, is given by

$$\min_u \lambda \|\mathrm{D}u\|_{p,1} + \|u - u^\diamond\|_1. \tag{2.8}$$

This model has many nice properties such as noise robustness and contrast invariance (Nikolova 2004, Chan and Esedoḡlu 2005). However, this does not come for free. While the ROF model still contains some regularity in the data term that can be exploited during optimization, the TV-ℓ_1 model is completely non-smooth and hence significantly more difficult to minimize.

2.2. Three introductory examples for image restoration

We now will present three prototypical examples of image restoration, to which we will frequently refer in the algorithmic parts of the paper.

Example 2.1 (ROF model). In the first example we consider standard image denoising using the ROF model (2.6) in the presence of Gaussian noise. Figure 2.1 shows the result of total variation based image denoising using this model. It is now well understood that efficient ways to solve this problem rely on convex duality (Chambolle and Lions 1995, Chan *et al.* 1999, Chambolle 2004*b*); for details on the particular algorithm used here, see Examples 4.8 and 5.6.

Figure 2.1(a) shows the original input image of size 360×270 pixels and intensity values in the range $[0, 1]$. Figure 2.1(b) shows its noisy variant, obtained by adding Gaussian noise of standard deviation $\sigma = 0.1$.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) original image

(b) blurry and noisy image

(c) deblurred, $\lambda = 0$

(d) deblurred, $\lambda = 5 \times 10^{-4}$

Figure 2.2. An image deblurring problem. (a) Original image, and (b) blurry and noisy image (Gaussian noise with standard deviation $\sigma = 0.01$) together with the known blur kernel. (c, d) Image deblurring without ($\lambda = 0$) and with ($\lambda = 5 \times 10^{-4}$) total variation regularization. Observe the noise amplification when there is no regularization.

Figure 2.1(c) shows the result obtained by minimizing the ROF model using the FISTA algorithm (Algorithm 5). We used isotropic total variation ($p = 2$) and we set the regularization parameter $\lambda = 0.1$. Observe that the ROF model successfully removes the noise from the image while preserving the main edges in the image. One can also observe that the ROF model is not very successful at reconstructing textured regions, as it favours piecewise constant images. State-of-the-art denoising methods will usually revert to non-local techniques that treat patches as a whole, allowing better representation of textures (Buades *et al.* 2005, Buades *et al.* 2011, Dabov *et al.* 2007). These approaches are not variational at first glance, but variants can be obtained by alternating minimization of non-local energies (Peyré *et al.* 2008, Arias *et al.* 2011).

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Example 2.2 (TV-deblurring). In this second example we assume that the observed blurry image $u^\diamond$ has been obtained by convolving the unknown image u with a two-dimensional blur kernel a of size $k \times l$ pixels. We can 'deblur' the given image by minimizing the model (2.7) with $Au = a * u$. If we choose $\lambda = 0$ in (2.7), then unless the original image $u^\diamond$ has no noise at all, it is well known that the noise will be amplified by the deconvolution process and ruin the quality of the deconvolution.

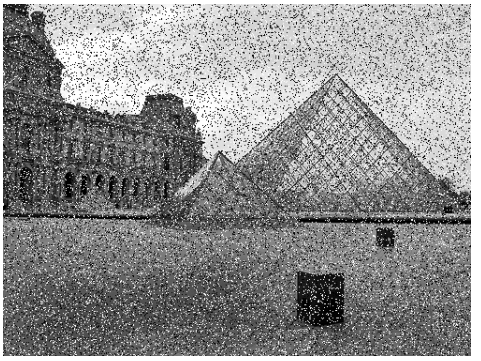
Figure 2.2 shows an example of image deblurring with known blur kernel. Figure 2.2(a) shows the original image of size 317×438 pixels and intensity values in the range $[0, 1]$. Figure 2.2(b) shows the blurry image together with the blur kernel of size 31×31 pixels. The blurry image has been further degraded by adding zero-mean Gaussian noise with standard deviation 0.01. Moreover, to get rid of unwanted boundary effects, we modified the input image by setting its intensity values to its average values at the image boundaries. This allows us to approximately assume periodic boundary conditions and hence to use a fast Fourier transform (FFT) to compute the convolution. Another way to deal with the boundary, which works better but is computationally more expensive, is suggested in Almeida and Figueiredo (2013).

Figure 2.2(c) shows the deblurred image using no regularization ($\lambda = 0$) and Figure 2.2(d) the deblurred image using the total variation regularized deblurring model. The regularization parameter was set to $\lambda = 5 \times 10^{-4}$. Observe that the regularization is essential to reduce the noise in the deblurred image. This particular example has been computed using the PDHG algorithm (Algorithm 6); see also Example 5.7 for details. Note that when the blur kernel is also unknown, the problem becomes non-convex and hence significantly more complex to solve (Levin, Weiss, Durand and Freeman 2011).

Example 2.3 (TV-ℓ_1 model). In this third example we consider image restoration in the presence of salt-and-pepper noise. For this we utilize the TV-ℓ_1 model (2.8). Figure 2.3 shows an example where the TV-ℓ_1 model can successfully denoise an image of size 375×500 pixels that has been degraded by adding 20% salt-and-pepper noise. The intensity values of the input image are again in the range $[0, 1]$. For comparison we also show the results of the ROF model (2.6) for this example. For the TV-ℓ_1 model the regularization parameter was set to $\lambda = 0.6$; for ROF, the regularization parameter was set to $\lambda = 0.25$. It can be seen that the results of the ROF model are significantly inferior, since the quadratic data term of the ROF model does not fit the distribution of the salt-and-pepper noise at all well. The example was computed again using the PDHG algorithm (Algorithm 6); see also Example 5.8 for details.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) original image



(b) noisy image

(c) TV-ℓ_1

(d) ROF

Figure 2.3. Denoising an image containing salt-and-pepper noise. (a) Original image, and (b) noisy image that has been degraded by adding 20% salt-and-pepper noise. (c) Denoised image obtained from the TV-ℓ_1 model, and (d) result obtained from the ROF model.

3. Notation and basic notions of convexity

We recall some basic notions of convexity, and introduce our notation. Throughout the paper, at least in the theoretical parts, $\mathcal{X}$ (and $\mathcal{Y}$) is a Hilbert or Euclidean space endowed with a norm $\|\cdot\| = \langle\cdot,\cdot\rangle^{1/2}$. The results in this section and the next should usually be understood in finite dimensions, but most of them do not depend on the dimension, and often hold in a Hilbert space. If M is a bounded positive definite symmetric operator, we define $\|x\|_M = \langle Mx, x\rangle^{1/2}$, which in finite-dimensional space is a norm equivalent to $\|x\|$.

In two-dimensional image processing we usually consider norms acting on images u defined on a regular Cartesian grid of $m \times n$ pixels. When the pixels are scalar-valued, that is, $u_{i,j} \in \mathbb{R}$, the image can also be written in the form $u = (u_{1,1}, \dots, u_{m,n}) \in \mathbb{R}^{m\times n}$.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

A p-vector norm acting on the image is hence given by

$$\|u\|_p = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |u_{i,j}|^p \right)^{1/p}.$$

When the pixels of an image $\mathbf{u}$ of size $m \times n$ pixels are vector-valued, we will adopt the notation $\mathbf{u} = (\mathbf{u}_{1,1}, \ldots, \mathbf{u}_{m,n}) \in \mathbb{R}^{m \times n \times r}$, with bold-font variables $\mathbf{u}_{i,j} \in \mathbb{R}^r$ referring to the vector-valued pixel. In such images we will consider mixed p, q-vector norms which are given by

$$\|\mathbf{u}\|_{p,q} = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |\mathbf{u}_{i,j}|_p^q \right)^{1/q},$$

with $|\mathbf{u}_{i,j}|_p = (\sum_{k=1}^{r} |u_{i,j,k}|^p)^{1/p}$ denoting the p-vector norm acting on the single pixels. Similarly, if the pixels are matrix-valued (or tensor-valued), that is, $\mathbf{U}_{i,j} \in \mathbb{R}^{r \times s}$, we have $\mathbf{U} = (\mathbf{U}_{1,1}, \ldots, \mathbf{U}_{m,n}) \in \mathbb{R}^{m \times n \times r \times s}$, and we will consider matrix norms, acting on the single pixels $\mathbf{U}_{i,j}$.

3.1. Convex functions

An extended real valued function $f : \mathcal{X} \to [-\infty, +\infty]$ is said to be *convex* if and only if its *epigraph*

$$\operatorname{epi} f := \{(x, \lambda) \in \mathcal{X} \times \mathbb{R} : \lambda \geq f(x)\}$$

is a convex set, that is, if when $\lambda \geq f(x)$, $\mu \geq f(y)$, and $t \in [0, 1]$, we have $t\lambda + (1-t)\mu \geq f(tx + (1-t)y)$.[3] It is *proper* if it is not identically $+\infty$ and nowhere $-\infty$: in this case, it is convex if and only if, for all $x, y \in \mathcal{X}$ and $t \in [0, 1]$,

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y).$$

It is *strictly convex* if the above inequality is strict whenever $x \neq y$ and $0 < t < 1$. It is *lower semi-continuous* (l.s.c.) if, for all $x \in \mathcal{X}$, if $x_n \to x$, then

$$f(x) \leq \liminf_{n \to \infty} f(x_n).$$

A trivial but important example is the *characteristic function* or *indicator function* of a set C:

$$\delta_C(x) = \begin{cases} 0 & \text{if } x \in C, \\ +\infty & \text{else,} \end{cases}$$

[3] This definition avoids the risky expression $(+\infty) + (-\infty)$; see for instance Rockafellar (1997, Section 4).

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

which is convex, l.s.c., and proper when C is convex, closed and non-empty. The minimization of such functions will allow us to easily model convex constraints in our problems.

3.2. Subgradient

Given a convex, extended real valued, l.s.c. function $f : \mathcal{X} \to [-\infty, +\infty]$, we recall that its subgradient at a point x is defined as the set

$$\partial f(x) := \{p \in \mathcal{X} : f(y) \geq f(x) + \langle p, y - x\rangle \text{ for all } y \in \mathcal{X}\}.$$

An obvious remark which stems from the definition is that this notion allows us to generalize Fermat's stationary conditions ($\nabla f(x) = 0$ if x is a minimizer of f) to non-smooth convex functions: we indeed have

$$x \in \mathcal{X} \text{ is a global minimizer of } f \text{ if and only if } 0 \in \partial f(x). \tag{3.1}$$

The function is *strongly convex* or 'μ-convex' if in addition, for $x, y \in \mathcal{X}$ and $p \in \partial f(x)$, we have

$$f(y) \geq f(x) + \langle p, y - x\rangle + \frac{\mu}{2}\|y - x\|^2$$

or, equivalently, if $x \mapsto f(x) - \mu\|x\|^2/2$ is also convex. It is then, obviously, strictly convex as it satisfies

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y) - \mu\frac{t(1-t)}{2}\|y - x\|^2 \tag{3.2}$$

for any x, y and any $t \in [0, 1]$. A trivial but important remark is that if f is strongly convex and x is a minimizer, then we have (since $0 \in \partial f(x)$)

$$f(y) \geq f(x) + \frac{\mu}{2}\|y - x\|^2$$

for all $y \in \mathcal{X}$.

The *domain* of f is the set $\operatorname{dom} f = \{x \in \mathcal{X} : f(x) < +\infty\}$, while the domain of ∂f is the set $\operatorname{dom} \partial f = \{x \in \mathcal{X} : \partial f(x) \neq \emptyset\}$. Clearly $\operatorname{dom} \partial f \subset \operatorname{dom} f$; in fact if f is convex, l.s.c. and proper, then $\operatorname{dom} \partial f$ is dense in $\operatorname{dom} f$ (Ekeland and Témam 1999). In finite dimensions, one can show that for a proper convex function, $\operatorname{dom} \partial f$ contains at least the relative interior of $\operatorname{dom} f$ (that is, the interior in the vector subspace which is generated by $\operatorname{dom} f$).

3.3. Legendre–Fenchel conjugate

To any function $f : \mathcal{X} \to [-\infty, +\infty]$ one can associate the *Legendre–Fenchel conjugate* (or convex conjugate)

$$f^*(y) = \sup_{x \in \mathcal{X}} \langle y, x\rangle - f(x) \tag{3.3}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

which, as a supremum of linear and continuous functions, is obviously convex and lower semi-continuous. The *biconjugate* f^{**} is then the largest convex l.s.c. function below f (from the definition it is easy to see that $f^{**} \leq f$); in particular, if f is already convex and l.s.c., we have $f^{**} = f$. This is a consequence of the convex separation theorem (a corollary of the Hahn–Banach theorem), which is a difficult result in general (see Brézis 1983 for an introduction to convex duality in infinite dimensions which includes a detailed proof of this result), but it is a trivial consequence of the projection onto closed convex sets in Euclidean or Hilbert spaces.

By definition, we see that x realizes the sup in (3.3) if and only if $y \in \partial f(x)$, and we have $f(x) + f^*(y) = \langle y, x\rangle$. In this case we easily deduce that $f^{**}(x) = f(x) = \langle y, x\rangle - f^*(x)$, so that in particular, y realizes the sup which defines $f^{**}(x)$. Also, it follows that $x \in \partial f^*(y)$. We deduce the celebrated Legendre–Fenchel identity:

$$y \in \partial f(x) \Leftrightarrow x \in \partial f^*(y) \Leftrightarrow f(x) + f^*(y) = \langle y, x\rangle. \tag{3.4}$$

In particular, ∂g and ∂g^* are inverses. From the definition, it is clear that the subgradient of a convex function is a *monotone operator*, that is, it satisfies

$$\langle p - q, x - y\rangle \geq 0 \quad \text{for all } (x, y) \in \mathcal{X}^2,\ p \in \partial f(x),\ q \in \partial f(y),$$

while it is *strongly monotone* if f is strongly convex:

$$\langle p - q, x - y\rangle \geq \mu\|x - y\|^2 \quad \text{for all } (x, y) \in \mathcal{X}^2,\ p \in \partial f(x),\ q \in \partial f(y).$$

An important remark is that a convex l.s.c. function f is μ-strongly convex if and only if its conjugate f^* is C^1 with $(1/\mu)$-Lipschitz gradient. In fact, f is μ-strongly convex if and only if, for any $x \in \mathcal{X}$ and $p \in \partial f(x)$, the 'parabola'

$$y \mapsto f(x) + \langle p, y - x\rangle + \frac{\mu}{2}\|y - x\|^2$$

touches the graph of f at x from below. But then, fairly simple computations show that the graph of f^* is touched from above at p by the conjugate parabola

$$q \mapsto \langle q, x\rangle + \frac{1}{2\mu}\|q - p\|^2 - f(x) = f^*(p) + \langle q - p, x\rangle + \frac{1}{2\mu}\|q - p\|^2,$$

which is equivalent to saying that $x = \nabla f^*(p)$ and (if this holds at all p) that $p \mapsto x = \nabla f^*(p)$ is $(1/\mu)$-Lipschitz. Observe that in this case, the strong monotonicity of ∂f also reads

$$\langle p - q, \nabla f^*(p) - \nabla f^*(q)\rangle \geq \mu\|\nabla f^*(p) - \nabla f^*(q)\|^2,$$

which expresses that ∇f^* is a μ-*co-coercive* monotone operator: in general the gradient of a convex function with L-Lipschitz gradient is $(1/L)$-co-coercive.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

We must mention here that subgradients of convex l.s.c. functions are only a particular class of *maximal monotone operators*, which are multivalued operators $T : \mathcal{X} \to \mathcal{P}(\mathcal{X})$ such that

$$\langle p - q, x - y \rangle \geq 0 \quad \text{for all } (x, y) \in \mathcal{X}^2,\ p \in Tx,\ q \in Ty \tag{3.5}$$

and whose graph $\{(x, p) : x \in Tp\} \subset \mathcal{X} \times \mathcal{X}$ is maximal (with respect to inclusion) in the class of graphs of operators which satisfy (3.5). Strongly monotone and co-coercive monotone operators are defined accordingly. It is also almost obvious from the definition that any maximal monotone operator T has an inverse T^{-1} defined by $x \in T^{-1}p \Leftrightarrow p \in Tx$, which is also maximal monotone. The operators ∂f and ∂f^* are inverse in this sense. Examples of maximal monotone operators which are not subgradients of a convex function are given by skew-symmetric operators. See, for instance, Brézis (1973) for a general study of maximal monotone operators in Hilbert spaces.

3.4. Proximal map and resolvent

Another important role in optimization is played by the so-called *proximal map* or *proximity operator* of a convex function defined as follows. If f is convex, proper and l.s.c., then clearly, for any x, there is a unique minimizer $\hat{y}$ to the strongly convex problem

$$\min_{y \in \mathcal{X}} f(y) + \frac{1}{2\tau}\|y - x\|^2, \tag{3.6}$$

which also satisfies

$$f(y) + \frac{1}{2\tau}\|y - x\|^2 \geq f(\hat{y}) + \frac{1}{2\tau}\|\hat{y} - x\|^2 + \frac{1}{2\tau}\|y - \hat{y}\|^2 \tag{3.7}$$

for any y (thanks to strong convexity). We let $\hat{y} := \operatorname{prox}_{\tau f}(x)$. It is easy to show that this defines a 1-Lipschitz, monotone operator, which is itself the gradient of a convex function. Basic subdifferential calculus (Rockafellar 1997) shows that

$$\partial f(\hat{y}) + \frac{\hat{y} - x}{\tau} \ni 0,$$

in other words $\hat{y} = (I + \tau\partial f)^{-1}x$ is given by the *resolvent* of the maximal monotone operator $\tau\partial f$ at x. In general it is shown that T is maximal monotone if and only if its resolvent $(I + T)^{-1}$ is well defined and single-valued; this is an important theorem due to Minty (1962). The resolvent is also a weak contraction, as well as a 'firmly non-expansive operator' (Bauschke, Moffat and Wang 2012), or equivalently a '1/2-averaged operator'; see Appendix A.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Playing with this expression and (3.4), we can easily deduce *Moreau's identity* (Moreau 1965)

$$x = (I + \tau\partial f)^{-1}(x) + \tau\left(I + \frac{1}{\tau}\partial f^*\right)^{-1}\left(\frac{x}{\tau}\right) = \mathrm{prox}_{\tau f}(x) + \tau\mathrm{prox}_{\frac{1}{\tau}f^*}\left(\frac{x}{\tau}\right), \tag{3.8}$$

which in fact holds for any maximal monotone operators T, T^{-1}. It shows in particular that if we know how to compute $\mathrm{prox}_{\tau f}$, then we also know how to compute $\mathrm{prox}_{f^*/\tau}$. Finally, we will sometimes let $\mathrm{prox}^M_{\tau f}(x)$ denote the proximity operator computed in the metric M, that is, the solution of

$$\min_{y\in\mathcal{X}} f(y) + \frac{1}{2\tau}\|y - x\|^2_M.$$

3.5. Fenchel–Rockafellar duality

We now introduce an essential notion in convex programming, that is, convex duality. This notion allows us to transform convex problems into other problems which sometimes have a nicer structure and are easier to tackle. A fairly extensive and very enlightening recent survey on duality for imaging and inverse problems can be found in Borwein and Luke (2015).

Consider the minimization problem

$$\min_{x\in\mathcal{X}} f(Kx) + g(x), \tag{3.9}$$

where

$$f : \mathcal{Y} \to (-\infty, +\infty], \quad g : \mathcal{X} \to (-\infty, +\infty]$$

are convex l.s.c. functions and $K : \mathcal{X} \to \mathcal{Y}$ is a bounded linear operator. Then, since $f = f^{**}$, one can write

$$\min_{x\in\mathcal{X}} f(Kx) + g(x) = \min_{x\in\mathcal{X}} \sup_{y\in\mathcal{Y}} \langle y, Kx\rangle - f^*(y) + g(x).$$

Under very mild conditions on f, g (such as $f(0) < \infty$ and g continuous at 0 (see *e.g.* Ekeland and Témam 1999, (4.21)); in finite dimensions it is sufficient to have a point x with both Kx in the relative interior of $\mathrm{dom}\, f$ and x in the relative interior of $\mathrm{dom}\, g$ (Rockafellar 1997, Corollary 31.2.1)), one can swap the min and sup in the relation above and write

$$\begin{aligned}
\min_x f(Kx) + g(x) &= \min_x \sup_y \langle y, Kx\rangle - f^*(y) + g(x) \\
&= \max_y \inf_x \langle y, Kx\rangle - f^*(y) + g(x) \qquad (3.10)\\
&= \max_y -f^*(y) - g^*(-K^*y). \qquad (3.11)
\end{aligned}$$

The last problem in this formula is the (Fenchel–Rockafellar) *dual problem*.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Under the assumptions above, it has at least a solution y^*. If x^* is any solution of the initial *primal* problem, then (x^*, y^*) is a saddle point of the primal–dual formulation: for any $(x, y) \in \mathcal{X} \times \mathcal{Y}$ we have

$$\mathcal{L}(x^*, y) \leq \mathcal{L}(x^*, y^*) \leq \mathcal{L}(x, y^*)$$

where

$$\mathcal{L}(x, y) := \langle y, Kx \rangle - f^*(y) + g(x) \tag{3.12}$$

denotes the *Lagrangian*. In particular, it satisfies

$$0 \in \partial g(x^*) + K^* y^*, \tag{3.13}$$
$$0 \in \partial f^*(y^*) - K x^*. \tag{3.14}$$

Observe that the *primal–dual gap*

$$\mathcal{G}(x, y) := f(Kx) + g(x) + f^*(y) + g^*(-K^* y) = \sup_{(x', y') \in \mathcal{X} \times \mathcal{Y}} \mathcal{L}(x, y') - \mathcal{L}(x', y),$$

which is always non-negative (even if the min and sup cannot be swapped), vanishes if and only if (x, y) is a saddle point.

Finally we remark that

$$T \begin{pmatrix} x \\ y \end{pmatrix} := \begin{pmatrix} \partial g(x) \\ \partial f^*(y) \end{pmatrix} + \begin{pmatrix} 0 & K^* \\ -K & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{3.15}$$

is a maximal monotone operator, being the sum of two maximal monotone operators, only one of which is a subgradient, and the conditions above can be written $T \binom{x^*}{y^*} \ni 0$.

Example 3.1 (dual of the ROF model). As an example, consider the minimization problem (2.6) above. This problem has the general form (3.9), with $x = u$, $K = \mathrm{D}$, $f = \lambda \|\cdot\|_{p,1}$ and $g = \|\cdot - u^\diamond\|^2/2$. Hence the dual problem (3.11) reads

$$\begin{aligned} \max_{\mathbf{p}} & -f^*(\mathbf{p}) - \left(\frac{1}{2} \|\mathrm{D}^* \mathbf{p}\|^2 - \langle \mathrm{D}^* \mathbf{p}, u^\diamond \rangle \right) \\ & = -\min_{\mathbf{p}} \left(f^*(\mathbf{p}) + \frac{1}{2} \|\mathrm{D}^* \mathbf{p} - u^\diamond\|^2 \right) + \frac{1}{2} \|u^\diamond\|^2, \end{aligned}$$

where $\mathbf{p} \in \mathbb{R}^{m \times n \times 2}$ is the dual variable. Equation (3.13) shows that the solution u of the primal problem is recovered from the solution $\mathbf{p}$ of the dual by letting $u = u^\diamond - \mathrm{D}^* \mathbf{p}$. One interesting observation is that the dual ROF model has almost exactly the same structure as the Lasso problem (2.2).

In this example, f is a norm, so f^* is the indicator function of the polar ball: in this case the dual variable has the structure $\mathbf{p} = (\mathbf{p}_{1,1}, \ldots, \mathbf{p}_{m,n})$,

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

where $\mathbf{p}_{i,j} = (p_{i,j,1}, p_{i,j,2})$ is the per pixel vector-valued dual variable, and therefore

$$f^*(\mathbf{p}) = \delta_{\{\|\cdot\|_{q,\infty} \le \lambda\}}(\mathbf{p}) = \begin{cases} 0 & \text{if } |\mathbf{p}_{i,j}|_q \le \lambda \text{ for all } i,j, \\ +\infty & \text{else,} \end{cases} \tag{3.16}$$

where q is the parameter of the polar norm ball which is defined via $1/p + 1/q = 1$. The most relevant cases are $p = 1$ or $p = 2$. In the first case we have $q = +\infty$, so the corresponding constraint reads

$$|\mathbf{p}_{i,j}|_\infty = \max\{|p_{i,j,1}|, |p_{i,j,2}|\} \le \lambda \quad \text{for all } i,j.$$

In the second case we have $q = 2$, and the corresponding constraint reads

$$|\mathbf{p}_{i,j}|_2 = \sqrt{p_{i,j,1}^2 + p_{i,j,2}^2} \le \lambda \quad \text{for all } i,j.$$

Of course, more complex norms can be used, such as the nuclear norm for colour images. In this case the per pixel dual variable $\mathbf{p}_{i,j}$ will be matrix-valued (or tensor-valued) and should be constrained to have its spectral (operator) norm less than λ, for all i,j. See Section 7.3 for an example and further details.

In practice, we will (improperly) use 'dual problem' to denote the minimization problem

$$\min\{\|\mathrm{D}^*\mathbf{p} - u^\diamond\|^2 : |\mathbf{p}_{i,j}|_q \le \lambda \text{ for all } i,j\}, \tag{3.17}$$

which is essentially a projection problem. For this problem, it is interesting to observe that the primal–dual gap

$$\begin{aligned} \mathcal{G}(u,\mathbf{p}) &= f(\mathrm{D}u) + \frac{1}{2}\|u - u^\diamond\|^2 + f^*(\mathbf{p}) + \frac{1}{2}\|\mathrm{D}^*\mathbf{p}\|^2 - \langle \mathrm{D}^*\mathbf{p}, u^\diamond\rangle \\ &= \lambda\|\mathrm{D}u\|_{p,1} + \delta_{\{\|\cdot\|_{q,\infty}\le\lambda\}}(\mathbf{p}) - \langle \mathbf{p}, \mathrm{D}u\rangle + \frac{1}{2}\|u^\diamond - \mathrm{D}^*\mathbf{p} - u\|^2 \end{aligned} \tag{3.18}$$

gives a bound on the ℓ_2-error $\frac{1}{2}\|u - u^*\|^2$, where $(u^*, \mathbf{p}^*)$ is a saddle point. More precisely, if we use both the strong convexity of the energy (with respect to u) and the strong convexity of the dual energy (with respect to $\mathrm{D}^*\mathbf{p}$) and recall that $u^* = u^\diamond - \mathrm{D}^*\mathbf{p}^*$ (so $\|\mathrm{D}^*\mathbf{p} - \mathrm{D}^*\mathbf{p}^*\|^2 = \|(u^\diamond - \mathrm{D}^*\mathbf{p}) - u^*\|^2$), we find that

$$\mathcal{G}(u,\mathbf{p}) \ge \frac{1}{2}\|u - u^*\|^2 + \frac{1}{2}\|(u^\diamond - \mathrm{D}^*\mathbf{p}) - u^*\|^2. \tag{3.19}$$

We can even provide a slightly finer criterion, since if we introduce the middle value

$$\tilde{u} := \frac{u + (u^\diamond - \mathrm{D}^*\mathbf{p})}{2},$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

then it follows from (3.19) that

$$\mathcal{G}(u,\mathbf{p}) \geq \|\tilde{u}-u^*\|^2 + \frac{1}{4}\|u^\diamond - \mathrm{D}^*\mathbf{p} - u\|^2.$$

Using (3.18) we obtain the error criterion

$$\lambda\|\mathrm{D}u\|_{p,1} + \delta_{\{\|\cdot\|_{q,\infty}\leq\lambda\}}(\mathbf{p}) - \langle \mathbf{p}, \mathrm{D}u\rangle + \frac{1}{4}\|u^\diamond - \mathrm{D}^*\mathbf{p} - u\|^2 \geq \|\tilde{u}-u^*\|^2. \tag{3.20}$$

This can be used to test the convergence of algorithms for this problem, at least when a dual variable $\mathbf{p}$ is correctly identified, since in that case, if u is not provided by the algorithm, one can let $u = \tilde{u} = u^\diamond - \mathrm{D}^*\mathbf{p}$. It also shows how to obtain, in a primal–dual method which provides both u and $\mathbf{p}$ with $u \neq \tilde{u}$, a slightly better estimate of the primal ℓ_2-error (and of the root-mean-square error $\|\tilde{u}-u^*\|/\sqrt{mn}$) than that given by the gap.

We will now describe, starting from the simplest, the first-order optimization methods that can be implemented to solve the problems described so far and a few others that will be introduced in Section 7.

4. Gradient methods

The first family of methods we are going to describe is that of first-order gradient descent methods. It might seem a bit strange to introduce such simple and classical tools, which might be considered outdated. However, as mentioned in the Introduction, the most efficient way to tackle many simple problems in imaging is via elaborate versions of plain gradient descent schemes. In fact, as observed in the 1950s, such methods can be considerably improved by adding inertial terms or performing simple over-relaxation steps (or less simple steps, such as Chebyshev iterations for matrix inversion: Varga 1962), line-searches, or more elaborate combinations of these, such as conjugate gradient descent; see for instance Polyak (1987, Section 3.2) or Bertsekas (2015, Section 2.1). Also, if second-order information is available, Newton's method or quasi-Newton variants such as the (l-)BFGS algorithm (Byrd, Lu, Nocedal and Zhu 1995) can be used, and are known to converge very fast. However, for medium/large non-smooth problems such as those described above, such techniques are not always convenient. It is now acknowledged that, if not too complex to implement, then simpler iterations, which require fewer operations and can sometimes even be parallelized, will generally perform better for a wide class of large-dimensional problems, such as those considered in this paper.

In particular, first-order iterations can be accelerated by many simple tricks such as over-relaxation or variable metrics – for instance Newton's method – but this framework can be transferred to fairly general schemes (Vũ 2013*b*, Combettes and Vũ 2014), and since the seminal contribution

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 1 Gradient descent (GD) with fixed step.

Choose $x^0 \in \mathcal{X}$
for all $k \geq 0$ **do**

$$x^{k+1} = x^k - \tau \nabla f(x^k). \tag{4.1}$$

end for

of Nesterov (1983) it has been understood that some of the over-relaxation techniques developed for matrix inversion can be adapted to the non-linear setting, providing efficient first-order schemes for non-linear or non-smooth minimization problems. Let us start with the simplest approach and show how it can be improved.

4.1. Gradient descent

We therefore start by describing gradient descent methods, and we will see that these are sufficient to provide efficient methods for solving simple problems such as the Lasso (2.2) or dual ROF (3.17) problems.

Assume we need to find a minimizer of a convex function f, that is, to solve

$$\min_{x \in \mathcal{X}} f(x), \tag{4.2}$$

and let us first assume that f is differentiable. Then, the most straightforward approach to solving the problem is to implement a gradient descent scheme with fixed step size $\tau > 0$: see Algorithm 1. The major issue is that this will typically not work if f is not sufficiently smooth. The natural assumption is that ∇f is Lipschitz with some constant L, and $0 < \tau L < 2$. If τ is too large, this method will oscillate: if for instance $f(x) = x^2/2$, then $x^{k+1} = (1-\tau)x^k$, and it is obvious that this recursion converges if and only if $\tau < 2$. On the other hand, a Taylor expansion shows that

$$f(x - \tau \nabla f(x)) \leq f(x) - \tau \left(1 - \frac{\tau L}{2}\right) \|\nabla f(x)\|^2,$$

so that if $\tau < 2/L$, then we see both that $f(x^k)$ is a strictly decreasing sequence (unless $\nabla f(x^k) = 0$ at some point) and that $\sum_k \|\nabla f(x^k)\|^2 < +\infty$ if f is bounded from below. If f is, in addition, coercive (with bounded level sets), it easily follows in finite dimensions that $f(x^k)$ converges to a critical value and that every converging subsequence of $(x^k)_{k \geq 0}$ goes to a critical point. If f is convex, then $x \mapsto x - \tau \nabla f(x)$ is also a (weak) contraction, which shows that $\|x^k - x^*\|$ is also non-increasing, for any minimizer[4] x^*

[4] We shall always assume the existence of at least one minimizer, here and elsewhere.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

of f. In this case we can deduce the convergence of the whole sequence $(x^k)_k$ to a solution, if $0 < \tau < 2/L$. In fact, this is a particular case of the fairly general theory of *averaged operators*, for which such iterations converge: see Theorem A.1 in the Appendix for details and references.

4.2. Implicit gradient descent and the proximal-point algorithm

The above analysis breaks down if ∇f is not Lipschitz, as clearly it will be much harder to understand how f and its gradient behave at the new point given by (4.1). One typical workaround is to use varying steps that converge to zero (see Section 4.3), but this leads to very slow algorithms. Another one is to try to implement an *implicit gradient descent* where the iteration (4.1) is replaced with

$$x^{k+1} = x^k - \tau \nabla f(x^{k+1}). \tag{4.3}$$

Now the question is how to implement this iteration. We clearly see that if such an x^{k+1} exists, then it satisfies

$$\nabla f(x^{k+1}) + \frac{x^{k+1} - x^k}{\tau} = 0,$$

so it is a critical point of the function

$$x \mapsto f(x) + \frac{\|x - x^k\|^2}{2\tau}. \tag{4.4}$$

If, in addition, f is convex and l.s.c. (rather than C^1), then this critical point is the unique minimizer of (4.4), that is, the proximal map of τf evaluated at x^k (see Section 3.4). We will say that the convex function f is *simple* if the proximal maps τf, $\tau > 0$, can be easily evaluated. An important observation is that its definition does not require any smoothness of f. Now consider the function

$$\bar{x} \mapsto f_\tau(\bar{x}) := \min_{x \in \mathcal{X}} f(x) + \frac{\|x - \bar{x}\|^2}{2\tau}, \tag{4.5}$$

which is the *Moreau–Yosida regularization* of f with parameter $\tau > 0$. It is a standard fact that $\nabla f_\tau(\bar{x})$ is a $(1/\tau)$-Lipschitz function whose gradient is given by

$$\nabla f_\tau(\bar{x}) = \frac{\bar{x} - \operatorname{prox}_{\tau f}(\bar{x})}{\tau}. \tag{4.6}$$

Indeed, for any x, y, letting $\xi = \operatorname{prox}_{\tau f}(x)$ and $\eta = \operatorname{prox}_{\tau f}(y)$,

$$\begin{aligned} f_\tau(y) &= f(\eta) + \frac{\|\eta - y\|^2}{2\tau} \\ &= f(\eta) + \frac{\|\eta - x\|^2}{2\tau} + \left\langle \frac{x - \eta}{\tau}, y - x \right\rangle + \frac{\|x - y\|^2}{2\tau} \end{aligned}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

$$\geq f(\xi) + \frac{\|\xi - x\|^2}{2\tau} + \left\langle \frac{x-\xi}{\tau}, y - x \right\rangle + \left\langle \frac{\xi - \eta}{\tau}, y - x \right\rangle + \frac{\|\eta - \xi\|^2}{2\tau} + \frac{\|x - y\|^2}{2\tau}$$
$$= f_\tau(x) + \left\langle \frac{x-\xi}{\tau}, y - x \right\rangle + \frac{\tau}{2}\left\| \frac{y-\eta}{\tau} - \frac{x-\xi}{\tau} \right\|^2,$$

which actually shows that $(x - \xi)/\tau$ is a subgradient of f_τ at x. In the third line we have used the fact that ξ is the minimizer of a $(1/\tau)$-strongly convex problem. The last term in this equation expresses the fact that the map $\bar{x} \mapsto (\bar{x} - \mathrm{prox}_{\tau f}(\bar{x}))/\tau$ is τ-co-coercive, which implies that it is also $(1/\tau)$-Lipschitz, as claimed.

Now we can rewrite (4.6) in the form

$$\mathrm{prox}_{\tau f}(\bar{x}) = \bar{x} - \tau \nabla f_\tau(\bar{x}),$$

which shows that an iteration of *implicit* gradient descent for f, which in its general form reads

$$x^{k+1} = \mathrm{prox}_{\tau f}(x^k) = (I + \tau \partial f)^{-1}(x^k) = x^k - \tau \nabla f_\tau(x^k), \tag{4.7}$$

is exactly the same as an iteration of *explicit* gradient descent for f_τ, with step τ (which is admissible since ∇f_τ is $(1/\tau)$-Lipschitz). As it is obvious from its definition (4.5) that both f and f_τ have the same set of minimizers, the convergence theory for the implicit algorithm is a simple corollary of the convergence theory for explicit gradient descent with fixed step. Moreover, as we know we can take slightly larger steps (we can let $x^{k+1} = x^k - \sigma \nabla f_\tau(x^k)$ for any choice of $\sigma < 2\tau$), then we also deduce the convergence for the iterations:

$$x^{k+1} = (1 + \theta)\mathrm{prox}_{\tau f}(x^k) - \theta x^k, \tag{4.8}$$

for any $\theta \in (-1, 1)$. These elementary remarks are in fact at the heart of the rich theory of contraction semigroups in Hilbert and Banach spaces, when applied to more general monotone or accretive operators (indeed, (4.6) is exactly the Yosida regularization of the subgradient operator ∂f). See for instance Brézis (1973).

In optimization theory, the extension of these remarks to a general monotone operator is known as the 'proximal-point algorithm' (PPA), introduced in Martinet (1970). The general form of the PPA algorithm is the iteration

$$x^{k+1} = (I + \tau_k T)^{-1} x^k \tag{4.9}$$

(with a possible relaxation as in (4.8)), which is shown to converge to a zero of T under certain conditions.[5] This method has been studied extensively

[5] This is an extension of Theorem A.1; see also the references cited there.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

since the 1970s (Martinet 1970, Rockafellar 1976), and in fact many of the methods we consider later on are special instances. Convergence proofs and rates of convergence can be found, for instance, in Brézis and Lions (1978) (these require $\sum_k \tau_k^2 = +\infty$, but $\sum_k \tau_k = +\infty$ is sufficient if $T = \partial f$); see also the work of Güler (1991) when $T = \partial f$. In fact some of the results mentioned in Section 4.7 below will apply to this method as a particular case, when $T = \partial f$, extending some of the results of Güler (1991).

Fairly general convergence rates for gradient methods are given in the rich book of Bertsekas (2015, Propositions 5.1.4, 5.1.5), depending on the behaviour of f near the set of minimizers. In the simplest case of the descent (4.1) applied to a function f with L-Lipschitz gradient, the convergence rate is found in many other textbooks (*e.g.* Nesterov 2004) and reads as follows.

Theorem 4.1. Let $x^0 \in \mathcal{X}$ and x^k be recursively defined by (4.1), with $\tau \leq 1/L$. Then not only does $(x^k)_k$ converge to a minimizer, but the value $f(x^k)$ decays with the rate

$$f(x^k) - f(x^*) \leq \frac{1}{2\tau k}\|x^* - x^0\|^2,$$

where x^* is any minimizer of f. If in addition f is strongly convex with parameter $\mu_f > 0$, we have

$$f(x^k) - f(x^*) + \frac{1}{2\tau}\|x^k - x^*\|^2 \leq \omega^k \frac{1}{2\tau}\|x^0 - x^*\|^2,$$

where $\omega = (1 - \tau\mu_f) < 1$.

A short (standard) proof is given in Appendix B.

Remark 4.2. This form of the result is slightly suboptimal, allowing a very elementary proof in Appendix B. However, it can be checked that the first rate holds for larger steps $\tau < 2/L$, while the second can be improved by taking larger steps ($\tau = 2/(L + \mu_f)$), yielding linear convergence with a factor $\omega = (1 - \mu_f/L)/(1 + \mu_f/L)$; see for instance Nesterov (2004, Theorem 2.1.15). However, we will see very soon that this too can be improved.

Of course, the observations above show that similar rates will also hold for the implicit form (4.7): indeed, recalling that $f_\tau(x^*) = f(x^*)$ for any $\tau > 0$, we have that a bound on $f_\tau(x^k) - f_\tau(x^*)$ is, by definition, also a bound on

$$f(x^{k+1}) - f(x^*) + \frac{\|x^{k+1} - x^k\|^2}{2\tau}.$$

We remark that in this implicit case it would seem that we only have to choose the largest possible τ to solve the minimization accurately. We will see further (Example 3.1) that in practice, we are not always free to choose the step or the metric which makes the algorithm actually implementable. In

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 2 Subgradient method (SGM).

Choose $x_0 \in \mathcal{X}$, $h_k > 0$ with $\sum_k h_k = +\infty$ and $\sum_k h_k^2 < +\infty$.
for all $k \geq 0$ **do**
Compute $g_k \in \partial f(x_k)$

$$x_{k+1} = x_k - h_k \frac{g_k}{\|g_k\|} \tag{4.10}$$

end for

other situations the choice of the step might eventually result in a trade-off between the precision of the computation, the overall rate and the complexity of one single iteration (which should also depend on τ).

4.3. Subgradient descent

Another way to implement a gradient scheme for a non-smooth convex objective is to perform a *subgradient descent*, that is, try to reduce the energy by following the direction of an arbitrarily chosen subgradient: see Algorithm 2. In general, this method performs poorly, as shown by Polyak (1987) and Nesterov (2004) for example, since the typical rate for such a method (which is also optimal: Nesterov 2004) is $O(1/\sqrt{k})$ for the best objective found after the kth iteration. However, if f is not 'simple' but ∂f is easy to compute, this might be an option, but it is preferable to try to use a splitting strategy as described in Section 4.7 and the following. A condition for convergence is that f should be M-Lipschitz (at least near the optimum), which is not too restrictive in finite dimensions since f is always locally Lipschitz in the interior of its domain.

The study of convergence of this algorithm is based on the following simple chain of (in)equalities: given x^* a minimizer, we have

$$\begin{aligned}
\|x_{k+1} - x^*\|^2 &= \left\| x_k - x^* - h_k \frac{g_k}{\|g_k\|} \right\|^2 \\
&= \|x_k - x^*\|^2 - 2\frac{h_k}{\|g_k\|}\langle x_k - x^*, g_k\rangle + h_k^2 \\
&\leq \|x_k - x^*\|^2 - 2\frac{h_k}{\|g_k\|}(f(x_k) - f(x^*)) + h_k^2
\end{aligned}$$

and hence (using $\|g_k\| \leq M$ and letting $x_k^o = \arg\min_{x_i, i \leq k} f(x_i)$)

$$f(x_k^o) - f(x^*) \leq M \frac{\sum_{i=0}^k h_i^2 + \|x_0 - x^*\|^2}{2\sum_{i=0}^k h_i},$$

which goes to 0 as $k \to \infty$ by assumption. If we now choose $h_i = C/\sqrt{k+1}$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(Nesterov 2004) for the first k iterations, then at iteration k we find

$$f(x_k^o) - f(x^*) \leq M\frac{C + \|x_0 - x^*\|^2}{2C\sqrt{k+1}}.$$

Clearly, this is much slower than the rate of descent with fixed steps which can be reached when ∇f is Lipschitz.

A variant proposed by Polyak (1987) consists in choosing $h_k = c_k(f(x^k) - f(x^*))/\|g_k\|^2$, with $c_k \in (\alpha, 2-\alpha)$ for some $\alpha > 0$. However, this requires knowledge, or a good estimate, of the optimal value $f(x^*)$. The rate for this approach is again $O(1/k)$.

Inexact variants, based on the notion of ε-subgradients (Rockafellar 1997), have also been introduced (Bertsekas and Mitter 1973, Polyak 1987). These have been studied recently by Benfenati and Ruggiero (2013) to tackle non-linear inverse problems in imaging (see also Bonettini, Benfenati and Ruggiero 2014). They have also been used by Bonettini and Ruggiero (2012) to reinterpret a primal–dual scheme of Zhu and Chan (2008) (see Section 5.1) and prove its convergence.

4.4. Lower bounds for smooth convex optimization

An important question is what is the best possible rate of convergence of a first-order method applied to a convex optimization problem. Of course, the answer depends on the properties of the function to minimize. An answer in the form of lower bounds has been given by Nemirovski and Yudin (1983), and is also found in Nesterov (2004). The idea is to consider a fairly general class of first-order methods where the iterates x^k are restricted to the subspace spanned by the gradients of earlier iterates, that is, for $k \geq 0$,

$$x^k \in x^0 + \operatorname{span}\{\nabla f(x^0), \nabla f(x^1), \dots, \nabla f(x^{k-1})\}, \tag{4.11}$$

where x^0 is an arbitrary starting point. Then, for $x \in \mathbb{R}^n$, we consider $L > 0$, $\mu \geq 0$, $1 \leq p \leq n$, the minimization of functions of the form

$$f(x) = \frac{L-\mu}{8}\left((x_1 - 1)^2 + \sum_{i=2}^{p}(x_i - x_{i-1})^2\right) + \frac{\mu}{2}\|x\|^2. \tag{4.12}$$

Starting from an initial point $x^0 = 0$, any first-order method of the class considered can transmit the information of the data term only at the speed of one coordinate index per iteration. This makes such problems very hard to solve by any first-order methods in the class (4.11). Indeed, if we start from $x^0 = 0$ in the above problem (whose solution is given by $x_k^* = 1$, $k = 1, \dots, p$, and 0 for $k > p$), then at the first iteration, only the first component x_1^1 will be updated (since $\partial f/\partial x_i(x^0) = 0$ for $i \geq 2$), and by induction we can check that $x_l^k = 0$ for $l \geq k+1$.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

For convenience we reproduce a variant of the results in Nesterov (2004) (where a slightly different function is used: see Theorems 2.1.7 and 2.1.13). If $\mu = 0$, using (possible translates of) the function (4.12), which is very ill conditioned (and degenerate if defined in dimension $n > p$), the following general lower bound for smooth convex optimization can be shown.

Theorem 4.3. For any $x^0 \in \mathbb{R}^n$, $L > 0$, and $k < n$ there exists a convex, continuously differentiable function f with L-Lipschitz-continuous gradient, such that for any first-order algorithm satisfying (4.11), we have

$$f(x^k) - f(x^*) \geq \frac{L\|x^0 - x^*\|^2}{8(k+1)^2}, \tag{4.13}$$

where x^* denotes a minimizer of f.

This particular bound is reached by considering the function in (4.12) with $p = k + 1$, and an appropriate change of variable which moves the starting point to the origin. Observe that the above lower bound is valid only if the number of iterates k is less than the problem size. We cannot improve this with a quadratic function, as the conjugate gradient method (which is a first-order method) is then known to find the global minimizer after at most n steps. But the practical problems we encounter in imaging are often so large that we will never be able to perform as many iterations as the dimension of the problem.

If choosing $\mu > 0$ so that the function (4.12) becomes μ-strongly convex, a lower bound for first-order methods is given in Theorem 2.1.13 of Nesterov (2004), which reads as follows.

Theorem 4.4. For any $x^0 \in \mathbb{R}^\infty \simeq \ell_2(\mathbb{N})$ and $\mu, L > 0$ there exists a μ-strongly convex, continuously differentiable function f with L-Lipschitz-continuous gradient, such that, for any algorithm in the class of first-order algorithms defined by (4.11), we have

$$f(x^k) - f(x^*) \geq \frac{\mu}{2}\left(\frac{\sqrt{q}-1}{\sqrt{q}+1}\right)^{2k}\|x^0 - x^*\|^2 \tag{4.14}$$

for all k, where $q = L/\mu \geq 1$ is the condition number, and x^* is the minimizer of f.

In finite dimensions, one can adapt the proof of Nesterov (2004) to show the same result for sufficiently small k, with respect to n. It is important to bear in mind that these lower bounds are inevitable for any first-order algorithm (assuming the functions are 'no better' than with L-Lipschitz gradient and μ-strongly convex). Of course, one could ask if these lower bounds are not too pessimistic, and whether such hard problems will appear in practice. We will indeed see that these lower bounds are highly relevant to our algorithms, and are observed when minimizing relatively simple problems

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 3 Accelerated gradient descent (AGD) with fixed step.

Choose $x^0 = x^{-1} = y^0 \in \mathcal{X}$, $\tau \leq 1/L$, and let $t_0 = 0$.
for all $k \geq 0$ **do**

$$\begin{aligned} t_{k+1} &= \frac{1+\sqrt{1+4t_k^2}}{2}, \\ y^k &= x^k + \frac{t_k - 1}{t_{k+1}}(x^k - x^{k-1}), \\ x^{k+1} &= y^k - \tau \nabla f(y^k). \end{aligned} \tag{4.15}$$

end for

such as the ROF model. Let us mention that many other types of interesting lower bounds can be found in the literature for most of the algorithmic techniques described in this paper, and a few others; see in particular the recent and fairly exhaustive study by Davis and Yin (2014*a*).

4.5. Accelerated gradient descent

Let us return to standard gradient descent. It turns out that the rates in Theorem 4.1 are suboptimal, in the sense that smaller upper bounds can be obtained, which (almost) match the lower bounds presented in the previous section. Accelerated variants of the gradient method for non-linear problems were first proposed by Nesterov (1983); see also Güler (1992) for the implicit form and variants, including inexact forms, and Salzo and Villa (2012) for a more general result. The method consists in simply performing a varying relaxation step at each iteration: see Algorithm 3.

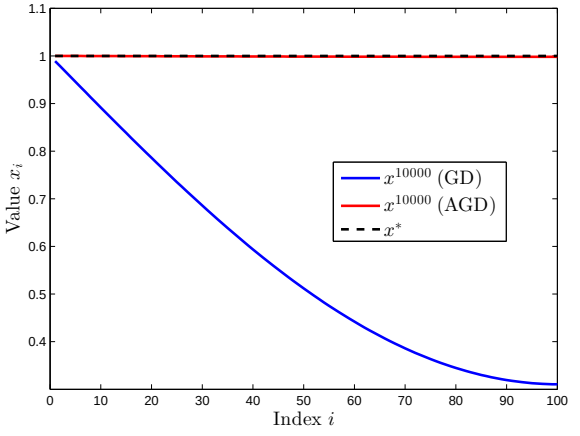
Theorem 4.5. Let $\{x^k\}$ be a sequence generated by the accelerated gradient descent (4.15). Then if x^* is a minimizer, we have

$$f(x^k) - f(x^*) \leq \frac{2}{\tau(k+1)^2}\|x^0 - x^*\|^2.$$

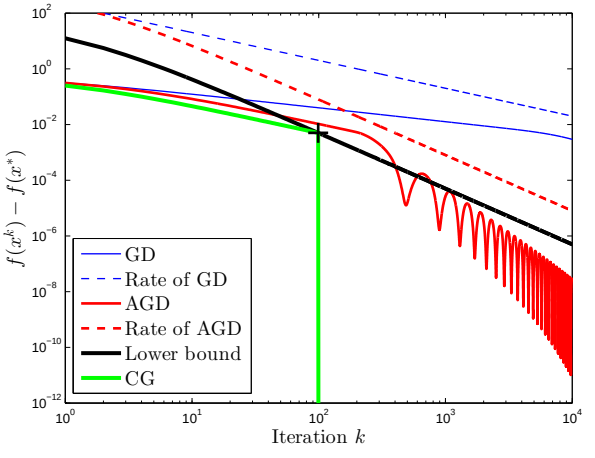
This rate is clearly better than the rate in Theorem 4.1, and, in fact, optimal when comparing to the lower bound in Theorem 4.3. We leave the case where f is strongly convex to Section 4.7 below, where we will present a more general result. Both are particular cases of Theorem B.1 in the Appendix.

Example 4.6 (minimizing the worst-case function). In this example we show the actual performance of gradient methods for the worst-case function presented in (4.12) using $p = n = 100$. Figure 4.1 compares the speed of convergence of gradient descent (GD), accelerated gradient descent (AGD),

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a)



(b)

Figure 4.1. Comparison of accelerated and non-accelerated gradient schemes. (a) Comparisons of the solutions x of GD and AGD after 10 000(!) iterations. (b) Rate of convergence for GD and AGD together with their theoretical worst-case rates, and the lower bound for smooth optimization. For comparison we also provide the rate of convergence for CG. Note that CG exactly touches the lower bound at $k = 99$.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

and conjugate gradient (CG), together with the lower bound for smooth optimization provided in (4.13). The results show that AGD is significantly faster than GD. For comparison we also applied CG, which is known to be an optimal method for quadratic optimization and provides convergence, in finitely many steps, to the true solution, in this case after at most $k = 100$ iterations. Observe that CG exactly touches the lower bound at $k = 99$ (black cross), which shows that the lower bound is sharp for this problem. Before and after $k = 99$, however, the lower bound is fairly pessimistic.

4.6. Descent on the Moreau–Yosida regularization

Let us consider a certain class of problems in which the objective function is the sum of a simple convex function and a quadratic function. This is the case for the dual of the ROF problem or the Lasso problem. We want to solve

$$\min_{x\in\mathcal{X}} f(x) := \frac{1}{2}\|Kx - x^\diamond\|^2 + g(x) \tag{4.16}$$

with g simple (*e.g.*, a characteristic function of a polar ball or an ℓ_1-norm). An important observation is that in this case the Moreau–Yosida regularization (4.5) of f is actually computable, provided that we choose the metric carefully. Let

$$M = \frac{1}{\tau}I - K^*K,$$

which is positive[6] if $\tau\|K\|^2 < 1$. Then the Moreau–Yosida regularization of f in the metric M is given by

$$f_M(\bar{x}) := \min_x \frac{1}{2}\|x - \bar{x}\|_M^2 + \frac{1}{2}\|Kx - x^\diamond\|^2 + g(x)$$

and the point $\hat{x} = \operatorname{prox}_f^M(\bar{x})$ which solves this problem is given by

$$\hat{x} = (I + \tau\partial g)^{-1}(\bar{x} - \tau K^*(K\bar{x} - x^\diamond)). \tag{4.17}$$

In other words, we can perform an implicit gradient descent (4.7) of f in the metric M. For the Lasso problem, this iteration is known as the 'iterative soft-thresholding' algorithm (Donoho 1995, Chambolle, DeVore, Lee and Lucier 1998, Daubechies, Defrise and De Mol 2004, Bect, Blanc-Féraud, Aubert and Chambolle 2004), as the proximity operator of $g = \|\cdot\|_1$ consists of a 'soft-thresholding' of the values. From (4.6), we see that $\nabla f_M(\bar{x}) = \hat{x} - \bar{x}$ (where the gradient is computed also in the metric M), and is (still in this metric) 1-Lipschitz. Therefore, we can solve the problem with a simple

[6] This point of view is a bit restrictive: it will be seen in Section 4.7 that one can also choose $\tau = 1/\|K\|^2$ – or even $\tau < 2/\|K\|^2$ for simple descent with fixed steps.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

gradient descent (4.1) on the function f_M (which is equivalent to (4.7) for f), or the accelerated descent described in Theorem 4.5.

It turns out that the operator in (4.17) can also be written (in the initial metric)

$$\hat{x} = \mathrm{prox}_{\tau g}\left(\bar{x} - \tau\nabla\left(\frac{1}{2}\|K\cdot - x^\diamond\|^2\right)(\bar{x})\right),$$

combining a step of implicit ('backward') gradient descent for g and a step of explicit ('forward') gradient descent for the smooth part $\frac{1}{2}\|K\cdot - x^\diamond\|^2$ of (4.16). This is a particular case of a more general gradient descent algorithm which mixes the two points of view explained so far, and which we describe in Section 4.7 below.

These first elementary convergence results can already be applied to quite important problems in imaging and statistics. We first consider plain gradient descent for the primal ROF problem and then show how we can use implicit descent to minimize the dual of the ROF problem (3.17), which has the same structure as the Lasso problem (2.2).

Example 4.7 (minimizing the primal ROF model). In this example we consider gradient descent methods to minimize the primal ROF model, in (2.6), for $p = 2$. As mentioned above, this will work only if the gradient of our energy is Lipschitz-continuous, which is not the case for (2.6). Hence we consider a smoothed version of the total variation, which is obtained by replacing the norm $\|\mathrm{D}u\|_{2,1}$, which is singular at 0, with a smoothed approximation; this means in practice that we solve a different problem, but we could theoretically estimate how far the solution to this problem is from the solution to the initial problem. A classical choice is

$$\sum_{i,j}\sqrt{\varepsilon^2 + (Du)_{i,j,1}^2 + (Du)_{i,j,2}^2},$$

where $\varepsilon > 0$ is a (usually small) parameter. While this approximation is C^∞, it tends to promote large gradients near the discontinuities of the image. A good alternative is the 'Huber regularizer'. Letting

$$h_\varepsilon(t) = \begin{cases} \dfrac{t^2}{2\varepsilon} & \text{if } t \le \varepsilon, \\ |t| - \dfrac{\varepsilon}{2} & \text{else,} \end{cases} \tag{4.18}$$

which is merely C^1 but smooths the absolute value function only locally around zero, we consider the following Huber-ROF problem:

$$\min_u f(u) = H_\varepsilon(\mathrm{D}u) + \frac{1}{2}\|u - u^\diamond\|^2, \tag{4.19}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

with

$$H_\varepsilon(\mathrm{D}u) = \lambda \sum_{i=1,j=1}^{m,n} h_\varepsilon\Big(\sqrt{(\mathrm{D}u)_{i,j,1}^2 + (\mathrm{D}u)_{i,j,2}^2}\Big). \tag{4.20}$$

Observe that (4.19) is strongly convex with parameter $\mu = 1$. Although we want to minimize the primal problem here, we remark that the dual of the Huber-ROF model is

$$\max_{\mathbf{p}} -\frac{1}{2}\|\mathrm{D}^*\mathbf{p}\|^2 + \langle \mathrm{D}^*\mathbf{p}, u^\diamond\rangle - H_\varepsilon^*(\mathbf{p}), \tag{4.21}$$

with

$$H_\varepsilon^*(\mathbf{p}) = \frac{\varepsilon}{2\lambda}\|\mathbf{p}\|^2 + \delta_{\{\|\cdot\|_{2,\infty}\le\lambda\}}(\mathbf{p}),$$

where $\delta_{\{\|\cdot\|_{2,\infty}\le\lambda\}}(\mathbf{p})$ denotes the characteristic function of the polar ball $\{\mathbf{p} : \|\mathbf{p}\|_{2,\infty} \le \lambda\}$ as in (3.16): it is simply the dual (3.17) of ROF, to which a small 'smoothing term' $\varepsilon/(2\lambda)\|\mathbf{p}\|^2$ has been added.

The gradient of (4.19) is computed as

$$\nabla f(u) = \mathrm{D}^*\tilde{\mathbf{p}} + u - u^\diamond,$$

where $\tilde{\mathbf{p}} = \nabla H_\varepsilon(Du)$, and it can be written as $\tilde{\mathbf{p}} = (\tilde{\mathbf{p}}_{1,1}, \dots, \tilde{\mathbf{p}}_{m,n})$, with $\tilde{\mathbf{p}}_{i,j}$ given by

$$\tilde{\mathbf{p}}_{i,j} = \frac{\lambda(\mathrm{D}u)_{i,j}}{\max\{\varepsilon, |(\mathrm{D}u)_{i,j}|_2\}}.$$

A simple computation shows that $\nabla f(u)$ is Lipschitz-continuous with parameter $L = 1 + (\|\mathrm{D}\|^2\lambda)/\varepsilon$, where $\|\mathrm{D}\| \le \sqrt{8}$ is the operator norm of D; see (2.5).

It can be observed that the auxiliary variables $\tilde{\mathbf{p}}$ are feasible (that is, $\delta_{\{\|\cdot\|_{2,\infty}\le\lambda\}}(\tilde{\mathbf{p}}) = 0$) dual variables, as by definition they satisfy (3.14). Hence we can use these expressions to compute the primal–dual gap (3.18) (where the regularizer and its conjugate now need to be replaced with $H_\varepsilon(\mathrm{D}u)$ and $H_\varepsilon^*(\mathbf{p})$):

$$\mathcal{G}(u, \tilde{\mathbf{p}}) = H_\varepsilon(\mathrm{D}u) + \frac{\varepsilon}{2\lambda}\|\tilde{\mathbf{p}}\|^2 - \langle \tilde{\mathbf{p}}, \mathrm{D}u\rangle + \frac{1}{2}\|u^\diamond - \mathrm{D}^*\tilde{\mathbf{p}} - u\|^2.$$

Using (3.20), we also obtain that

$$H_\varepsilon(\mathrm{D}u) + \frac{\varepsilon}{2\lambda}\|\tilde{\mathbf{p}}\|^2 - \langle \tilde{\mathbf{p}}, \mathrm{D}u\rangle + \frac{1}{4}\|u^\diamond - \mathrm{D}^*\tilde{\mathbf{p}} - u\|^2 \ge \|\tilde{u} - u^*\|^2,$$

where u^* is the solution of (4.19) and $\tilde{u} = (u + u^\diamond - \mathrm{D}^*\tilde{\mathbf{p}})/2$. We implement the gradient descent algorithm (4.1) using a constant step size $\tau = 2/(L + \mu)$ and apply the algorithm to Example 2.1. Figure 4.2 shows the convergence of the primal–dual gap using different values of ε. Since the objective function is smooth and strongly convex, the gradient descent

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

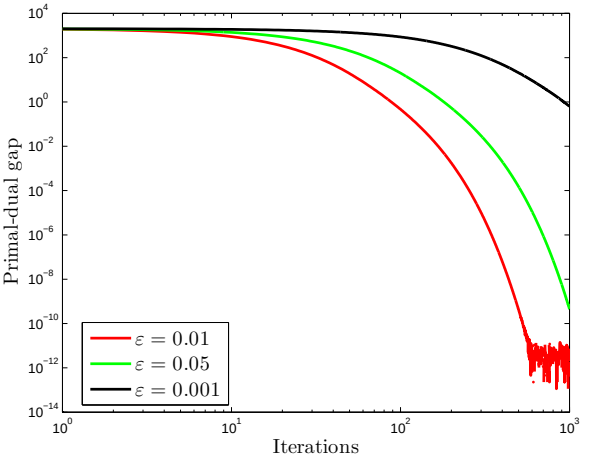


Figure 4.2. Minimizing the primal ROF model using smoothed (Huber) total variation applied to the image in Figure 2.1. The figure shows the convergence of the primal–dual gap using plain gradient descent for different settings of the smoothing parameter ε.

converges linearly. However, for smaller values of ε, where the smoothed ROF model approaches the original ROF model, the convergence of the algorithm becomes very slow. The next example shows that it is actually a better idea to minimize the dual of the ROF model.

Example 4.8 (minimizing the dual ROF model). Let us turn to the problem of minimizing the dual ROF model using the explicit representation of the Moreau–Yosida envelope. We consider (4.16) with $K = \mathrm{D}$ and $g = \delta_{\{\|\cdot\|_{2,\infty} \leq \lambda\}}$. The Moreau–Yosida regularization is given by

$$f_M(\bar{\mathbf{p}}) := \min_{\mathbf{p}} \frac{1}{2}\|\mathbf{p} - \bar{\mathbf{p}}\|_M^2 + \frac{1}{2}\|\mathrm{D}^*\mathbf{p} - u^\diamond\|^2 + \delta_{\{\|\cdot\|_{2,\infty} \leq \lambda\}}(\mathbf{p}), \tag{4.22}$$

with τ' such that $M = (1/\tau')\, I - \mathrm{D}\mathrm{D}^* > 0$, and the minimum of the right-hand side is attained for

$$\hat{\mathbf{p}} = \Pi_{\{\|\cdot\|_{2,\infty} \leq \lambda\}}(\bar{\mathbf{p}} - \tau'\mathrm{D}(\mathrm{D}^*\bar{\mathbf{p}} - u^\diamond)),$$

where $\Pi_{\{\|\cdot\|_{2,\infty} \leq \lambda\}}$ denotes the (pixelwise) orthogonal projection onto 2-balls with radius λ, that is, for each pixel i, j, the projection is computed by

$$\hat{\mathbf{p}} = \Pi_{\{\|\cdot\|_{2,\infty} \leq \lambda\}}(\tilde{\mathbf{p}}) \Leftrightarrow \hat{\mathbf{p}}_{i,j} = \frac{\tilde{\mathbf{p}}_{i,j}}{\max\{1, \lambda^{-1}|\tilde{\mathbf{p}}_{i,j}|_2\}}. \tag{4.23}$$

As shown before, the gradient in the M-metric is given by

$$\nabla f_M(\bar{\mathbf{p}}) = \bar{\mathbf{p}} - \hat{\mathbf{p}}. \tag{4.24}$$

The advantages of minimizing the dual ROF model, rather than the primal

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

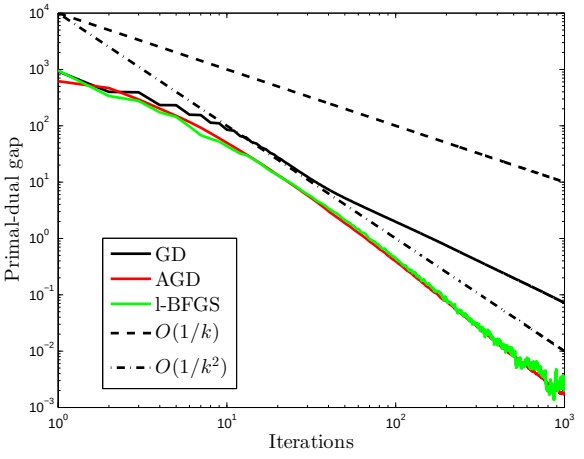


Figure 4.3. Comparison of different gradient-based methods applied to Moreau–Yosida regularization of the dual ROF model using the image in Figure 2.1. Accelerated gradient descent (AGD) and the quasi-Newton method (l-BFGS) are significantly faster than plain gradient descent (GD).

ROF model as in Example 4.7, are immediate. Thanks to the implicit smoothing of the Moreau–Yosida regularization, we do not need to artificially smooth the objective function and hence any gradient method will converge to the exact minimizer. Second, the step size of a gradient method will just depend on $\|\mathrm{D}\|$, whereas the step size of a gradient method applied to the primal ROF model is proportional to the smoothing parameter ε. We implement both a standard gradient descent (GD) with step size $\tau = 1.9$ and the accelerated gradient descent (AGD) with step size $\tau = 1$. The parameter τ' in the M-metric is set to $\tau' = 0.99/\|\mathrm{D}\|^2$.

Since we are dealing with a smooth, unconstrained optimization in (4.22), we can also try to apply a black-box algorithm, which only needs information about the gradients and the function values. A very popular algorithm is the limited memory BFGS quasi-Newton method (Byrd *et al.* 1995, Zhu, Byrd, Lu and Nocedal 1997, Morales and Nocedal 2011). We applied a 1-memory variant of the l-BFGS algorithm[7] to the Moreau–Yosida regularization of the dual ROF model and supplied the algorithm with function values (4.22) (using the correct values of $\hat{p}$) and gradients (4.24). The idea of using variable metric approaches to the Moreau–Yosida regularization of the operator has been investigated in many papers (Bonnans, Gilbert, Lemaréchal and Sagastizábal 1995, Burke and Qian 1999, Burke and Qian 2000) and can lead to very fast convergence under simple smoothness assumptions. However,

[7] We used S. Becker's MATLAB wrapper of the implementation at http://users.iems.northwestern.edu/~nocedal/lbfgsb.html.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 4 Forward–backward descent with fixed step.

Choose $x_0 \in \mathcal{X}$
for all $k \geq 0$ **do**

$$x^{k+1} = T_\tau x^k = \mathrm{prox}_{\tau g}(x^k - \tau \nabla f(x^k)). \tag{4.25}$$

end for

it is not always suitable or easily implementable for many of the problems we address in this paper.

The plot in Figure 4.3 represents the decay of the primal–dual gap (which bounds the energy and the ℓ_2-error) obtained from gradient descent (GD), accelerated gradient descent (AGD) and the limited memory BFGS quasi-Newton method (l-BFGS). It appears that the energy actually decreases faster for the accelerated method and the quasi-Newton method, with no clear advantage of one over the other (the first being of course simpler to implement). Also observe that both AGD and l-BFGS are only slightly faster than the lower bound $O(1/k^2)$ for smooth convex optimization. This shows that the dual ROF model is already quite a hard optimization problem. We should mention here that the idea of applying quasi-Newton methods to a regularized function as in this example has been recently extended to improve the convergence of some of the methods introduced later in this paper, namely the forward–backward and Douglas–Rachford splittings, with very interesting results: see Patrinos, Stella and Bemporad (2014) and Stella, Themelis and Patrinos (2016).

4.7. Forward–backward splitting

We can write problem (4.16) in the general form

$$\min_{x \in \mathcal{X}} F(x) := f(x) + g(x), \tag{4.26}$$

where g is, as before, a 'simple' convex l.s.c. function and f is a convex function with Lipschitz gradient. The basic idea of the forward–backward (FB) splitting scheme is to combine an explicit step of descent in the smooth part f with a implicit step of descent in g. We thus introduce the operator

$$\bar{x} \mapsto \hat{x} = T_\tau \bar{x} := \mathrm{prox}_{\tau g}(\bar{x} - \tau \nabla f(\bar{x})) = (I + \tau \partial g)^{-1}(\bar{x} - \tau \nabla f(\bar{x})). \tag{4.27}$$

Another name found in the literature (Nesterov 2013) is 'composite gradient' descent, as one may see $(\hat{x} - \bar{x})/\tau$ as a generalized gradient for F at $\bar{x}$ (in particular, note the analogy with (4.6)). The essential reason justifying this is that a fixed point $\hat{x} = \bar{x}$ will clearly satisfy the stationary condition $\nabla f(\bar{x}) + \partial g(\bar{x}) \ni 0$ of (4.26). Observe that in the particular case where $g = \delta_C$ is the characteristic function of a closed, convex set C, then

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

$\mathrm{prox}_{\tau g}(x)$ reduces to $\Pi_C(x)$ (the orthogonal projection onto C) and the mapping T_τ defines a projected gradient descent method (Goldstein 1964). See Algorithm 4.

The theoretical convergence rate of plain FB splitting descent is not very good, as one can simply show the same as for gradient descent.

Theorem 4.9. Let $x^0 \in \mathcal{X}$ and x^k be recursively defined by (4.25), with $\tau \le 1/L$. Then not only does x^k converge to a minimizer but we have the rates

$$F(x^k) - F(x^*) \le \frac{1}{2\tau k}\|x^* - x^0\|^2, \tag{4.28}$$

where x^* is any minimizer of f. If in addition f or g is strongly convex with parameters μ_f, μ_g (with $\mu = \mu_f + \mu_g > 0$), we have

$$F(x^k) - F(x^*) + \frac{1+\tau\mu_g}{2\tau}\|x^k - x^*\|^2 \le \omega^k \frac{1+\tau\mu_g}{2\tau}\|x^0 - x^*\|^2, \tag{4.29}$$

where $\omega = (1-\tau\mu_f)/(1+\tau\mu_g)$.

However, its behaviour is improved if the objective is smoother than actually known. Moreover, it is fairly robust to perturbations and can be over-relaxed; see in particular Combettes and Wajs (2005).

An 'optimal' accelerated version, generalizing Theorem 4.5, is also available for this method. This is introduced in Nesterov (2004) (for projected gradient descent). In the case $\mu = \mu_f + \mu_g = 0$, a more general algorithm, popularized under the name 'FISTA', is proposed in Beck and Teboulle (2009). The algorithm we present here unifies these approaches. The general iteration takes the form shown in Algorithm 5. In (4.35), we can assume $L > \mu_f$, and hence $\tau\mu_f < 1$; otherwise f is quadratic and the problem is trivial. We have the following result, which unifies Nesterov (2004) and Beck and Teboulle (2009). See also Nesterov (2005, 2013) and Tseng (2008) for more general variants that enjoy the same convergence rates.

Theorem 4.10. Assume $t_0 = 0$ and let x^k be generated by the algorithm, in either case $\mu = 0$ or $\mu > 0$. Then we have the decay rate

$$F(x^k) - F(x^*) \le \min\left\{(1+\sqrt{q})(1-\sqrt{q})^k, \frac{4}{(k+1)^2}\right\}\frac{1+\tau\mu_g}{2\tau}\|x^0 - x^*\|^2.$$

It must be mentioned that for $\mu = 0$, a classical choice for t_k is also $t_k = (k+1)/2$, which gives essentially the same rate. Variants of this choice which ensure, in addition, convergence of the iterates $(x_k)_k$ to a solution, are discussed in Chambolle and Dossal (2015). An important issue is the stability of these rates when the proximal operators can only be evaluated approximately: the situation here is worse than for the non-accelerated algorithm. Several papers address this issue and derive the corresponding rates, for example Schmidt, Roux and Bach (2011), Villa, Salzo, Baldassarre

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 5 FISTA (and variant)

Given $0 < \tau \leq 1/L$, let $q = \tau\mu/(1+\tau\mu_g) < 1$. Choose $x^0 = x^{-1} \in \mathcal{X}$, and $t_0 \in \mathbb{R}$, $0 \leq t_0 \leq 1/\sqrt{q}$.
for all $k \geq 0$ **do**

$$y^k = x^k + \beta_k(x^k - x^{k-1}) \tag{4.30}$$
$$x^{k+1} = T_\tau y^k = \mathrm{prox}_{\tau g}(y^k - \tau\nabla f(y^k)) \tag{4.31}$$

where, for $\mu = 0$,

$$t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2} \geq \frac{k+1}{2}, \tag{4.32}$$
$$\beta_k = \frac{t_k - 1}{t_{k+1}}, \tag{4.33}$$

and if $\mu = \mu_f + \mu_g > 0$,

$$t_{k+1} = \frac{1 - qt_k^2 + \sqrt{(1-qt_k^2)^2 + 4t_k^2}}{2}, \tag{4.34}$$
$$\beta_k = \frac{t_k - 1}{t_{k+1}} \frac{1 + \tau\mu_g - t_{k+1}\tau\mu}{1 - \tau\mu_f}. \tag{4.35}$$

end for

and Verri (2013) and Aujol and Dossal (2015); see also Güler (1992) and Salzo and Villa (2012) for the backward step only, and d'Aspremont 2008 in the smooth case. Naturally, a rate of convergence for the errors is required to obtain an improved global rate.

Proofs of both Theorems 4.9 and 4.10 are given in Appendix B, where more cases are discussed, including more possibilities for the choice of the parameters. They rely on the following essential but straightforward descent rule.[8] Let $\hat{x} = T_\tau \bar{x}$. Then, for all $x \in \mathcal{X}$,

$$F(x) + (1-\tau\mu_f)\frac{\|x-\bar{x}\|^2}{2\tau} \geq \frac{1-\tau L}{\tau}\frac{\|\hat{x}-\bar{x}\|^2}{2} + F(\hat{x}) + (1+\tau\mu_g)\frac{\|x-\hat{x}\|^2}{2\tau}. \tag{4.36}$$

In particular, if $\tau L \leq 1$,

$$F(x) + (1-\tau\mu_f)\frac{\|x-\bar{x}\|^2}{2\tau} \geq F(\hat{x}) + (1+\tau\mu_g)\frac{\|x-\hat{x}\|^2}{2\tau}. \tag{4.37}$$

[8] This rule – or some variant of it – is of course found in almost all papers on first-order descent methods.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

The proof is elementary, especially if we follow the lines of the presentation in Tseng (2008), in a more general setting. By definition, $\hat{x}$ is the minimizer of the $(\mu_g + (1/\tau))$-strongly convex function

$$x \mapsto g(x) + f(\bar{x}) + \langle \nabla f(\bar{x}), x - \bar{x} \rangle + \frac{\|x - \bar{x}\|^2}{2\tau}.$$

It follows that for all x (see (3.7))

$$\begin{aligned} F(x) + (1 - \tau\mu_f)\frac{\|x - \bar{x}\|^2}{2\tau} \\ \geq g(x) + f(\bar{x}) + \langle \nabla f(\bar{x}), x - \bar{x} \rangle + \frac{\|x - \bar{x}\|^2}{2\tau} \\ \geq g(\hat{x}) + f(\bar{x}) + \langle \nabla f(\bar{x}), \hat{x} - \bar{x} \rangle + \frac{\|\hat{x} - \bar{x}\|^2}{2\tau} + (1 + \tau\mu_g)\frac{\|x - \hat{x}\|^2}{2\tau}. \end{aligned}$$

But since ∇f is L-Lipschitz, $f(\bar{x}) + \langle \nabla f(\bar{x}), \hat{x} - \bar{x} \rangle \geq f(\hat{x}) - (L/2)\|\hat{x} - \bar{x}\|^2$, so equation (4.36) follows.

Remark 4.11. One can more precisely deduce from this computation that

$$\begin{aligned} F(x) + (1 - \tau\mu_f)\frac{\|x - \bar{x}\|^2}{2\tau} \\ \geq F(\hat{x}) + (1 + \tau\mu_g)\frac{\|x - \hat{x}\|^2}{2\tau} + \left(\frac{\|\hat{x} - \bar{x}\|^2}{2\tau} - D_f(\hat{x}, \bar{x}) \right), \end{aligned} \tag{4.38}$$

where $D_f(x, y) := f(x) - f(y) - \langle \nabla f(y), x - y \rangle \leq (L/2)\|x - y\|^2$ is the 'Bregman f-distance' from y to x (Bregman 1967). In particular, (4.37) holds once

$$D_f(\hat{x}, \bar{x}) \leq \frac{\|\hat{x} - \bar{x}\|^2}{2\tau},$$

which is always true if $\tau \leq 1/L$ but might also occur in other situations, and in particular, be tested 'on the fly' during the iterations. This allows us to implement efficient backtracking strategies of the type of Armijo (1966) (see Nesterov 1983, Nesterov 2013, Beck and Teboulle 2009) for the algorithms described in this section when the Lipschitz constant of f is not *a priori* known.

Remark 4.12. Observe that if $X \subset \mathcal{X}$ is a closed convex set containing the domain of F, and on which the projection Π_X can be computed, then the same inequality (4.37) holds if $\hat{x} = T_\tau \Pi_X \bar{x}$ (requiring only that ∇f is Lipschitz on X), provided $x \in X$; see Bonettini, Porta and Ruggiero (2015). This means that the same rates are valid if we replace (4.30) with

$$y^k = \Pi_X(x^k + \beta_k(x^k - x^{k-1})),$$

which is feasible if X is the domain of F.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Discussion

The idea of forward–backward splitting is very natural, and appears in many papers in optimization for imaging: it would not be possible to mention all the related literature. Historically, it is a generalization of projected gradient descent, which dates back at least to Goldstein (1964) (see Passty 1979, Lions and Mercier 1979, Fukushima and Mine 1981). For minimization problems, it can be viewed as successive minimizations of a parabolic upper bound of the smooth part added to the non-smooth part. It has been generalized, and popularized in the imaging community by Combettes and Wajs (2005), yet a few particular forms were already well known, such as iterative soft-thresholding for the Lasso problem (Daubechies *et al.* 2004). It is not always obvious how to choose parameters correctly when they are unknown. Several backtracking techniques will work, such as those of Nesterov (2013), for both the Lipschitz constants and strong convexity parameters; see also Nesterov (1983), Beck and Teboulle (2009) and Bonettini *et al.* (2015) for estimates of the Lipschitz constant.

For simpler problems such as Lasso (2.2), convergence of the iterates (more precisely of Ax^k) yields that after some time (generally unknown), the support $\{i : x_i^* = 0\}$ of the solution x^* should be detected by the algorithm (under 'generic' conditions). In that case, the objective which is solved becomes smoother than during the first iterations, and some authors have succeeded in exploiting this 'partial smoothness' to show better (linear) convergence of the FB descent (Bredies and Lorenz 2008, Grasmair, Haltmeier and Scherzer 2011, Liang, Fadili and Peyré 2014, Tao, Boley and Zhang 2015). Liang, Fadili and Peyré (2015) have extended this approach to the abstract setting of Appendix A, so that this remark also holds for some of the saddle-point-type algorithms introduced in Section 5 below.

Another interesting and alternative approach to convergence rates is to use the 'Kurdyka–Łojasiewicz' (KL) inequality, which in practice will bound a function of the distance of a point to the critical set by the norm of the (sub)gradient. As shown by Bolte, Daniilidis and Lewis (2006), such a property will hold for 'most' of the functions optimized in practice, including non-smooth functions, and this can lead to improved convergence rates for many algorithms (Attouch, Bolte and Svaiter 2013). It is also possible to derive accelerated schemes for problems with different types of smoothness (such as Hölder-continuous gradients); see Nesterov (2015).

Finally, a heuristic technique which often works to improve the convergence rate, when the objective is smoother than actually known, consists simply in 'restarting' the method after a certain number of iterations: in Algorithm 5 (for $\mu = 0$), we start with a new sequence $(t_k)_k$ letting $t_{\bar{k}} = 1$ for some sufficiently large $\bar{k}$. Ideally, we should restart when we are sure that the distance of $x^{\bar{k}}$ to the optimum x^* (unique if the objective is strongly convex) has shrunk by a given, sufficiently small factor (but the correspond-

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

ing value $\bar{k}$ depends on the strong convexity parameter). There is a simple way to implement such a scheme while still keeping the global $O(1/k^2)$ rate (it consists in adapting the idea of the 'Monotone FISTA' scheme: see Remark B.3). A rigorous justification of a restarting scheme is discussed in O'Donoghue and Candès (2015).

4.8. Extensions and variants

4.8.1. Mirror descent

A natural extension of the proximal descent methods consists in replacing the function $(2\tau)^{-1}\|y-x\|^2$ in (3.6) with other distances between y and x. There can be several good reasons for this.

- One may wish to use a (smooth) distance $d(y,x)$ which blows up when y reaches the boundary of certain constraints. This is the principle of barriers and penalty functions.
- The proximity operator of a function is not easily computable with the squared Euclidean distance but it is simple in some non-linear metrics.
- The ambient space $\mathcal{X}$ is neither a Hilbert space nor a Euclidean space, and we need to optimize a function whose gradient is Lipschitz with respect to some non-Hilbert space norm.

The 'mirror' descent algorithm was introduced by Nemirovski and Yudin (1983) as a tool for optimization in Banach spaces. It requires the introduction of an auxiliary convex function ψ whose gradient will act as a map between the space $\mathcal{X}$ and its dual $\mathcal{X}'$. In the Hilbert space case, $\psi(x) = \|x\|^2/2$ is the most natural choice, but there might be reasons to use other choices (whereas in Banach spaces it is not natural at all). The basic idea is to replace the gradient descent iteration (4.1) with

$$\nabla\psi(x^{k+1}) = \nabla\psi(x^k) - \tau\nabla f(x^k).$$

If we introduce the Bregman ψ-distance

$$D_\psi(x,y) = \psi(x) - \psi(y) - \langle\nabla\psi(y), x-y\rangle,$$

we readily see that it is equivalent to defining x^{k+1} as a point which minimizes

$$\min_x \frac{1}{\tau}D_\psi(x,x^k) + f(x^k) + \langle\nabla f(x^k), x-x^k\rangle, \tag{4.39}$$

that is, we find x^{k+1} by minimizing the linear approximation of f at x^k with some penalization of the distance between the two points. The natural 'mirror prox' alternative will consist in solving iteratively, if possible,

$$\min_x \frac{1}{\tau}D_\psi(x,x^k) + f(x) \tag{4.40}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

and defining x^{k+1} to be the solution. In general it is required that ψ be smooth and strongly convex with respect to the norm of the space $\mathcal{X}$, which is not necessarily Euclidean or Hilbertian. Convergence of these algorithms under various conditions on ψ are studied in a few important papers; see in particular the papers of Eckstein (1993), Teboulle (1992), Chen and Teboulle (1993), Kiwiel (1997), Beck and Teboulle (2003) and Auslender and Teboulle (2004). The extensive monograph by Ben-Tal and Nemirovski (2001)[9] presents many possible variants, with rates.

An important remark is that a non-linear variant of (4.37) is as easy to show in the non-linear case, since if $\hat{x}$ is the minimizer of

$$\min_x \frac{1}{\tau} D_\psi(x, \bar{x}) + f(x)$$

for some f and admissible ψ, it satisfies

$$\nabla\psi(\hat{x}) - \nabla\psi(\bar{x}) + \partial f(\hat{x}) \ni 0,$$

from which we deduce from simple computations that for any $x \in \mathcal{X}$,

$$\frac{1}{\tau} D_\psi(x, \bar{x}) + f(x) \geq \frac{1}{\tau} D_\psi(\hat{x}, \bar{x}) + f(\hat{x}) + \frac{1}{\tau} D_\psi(x, \hat{x}). \tag{4.41}$$

It is relatively easy to deduce basic rates of convergence for the mirror and mirror prox schemes from this inequality, in the same way as for the Hilbertian FB splitting.

Quite naturally, this can be generalized to the full forward–backward splitting, where now the problem is of the form (4.26), and the point $\hat{x}$ is obtained from $\bar{x}$ by solving

$$\min_{x \in \mathcal{X}} \frac{1}{\tau} D_\psi(x, \hat{x}) + \langle \nabla f(\bar{x}), x \rangle + g(x).$$

A non-linear analogue of (4.37) will easily follow from (4.41) and the Lipschitz property of ∇f, which reads

$$\|\nabla f(x) - \nabla f(y)\|_* \leq L\|x - y\| \quad \text{for all } x, y \in \mathcal{X},$$

where $\|\cdot\|_*$ is the norm in $\mathcal{X}'$ induced by the norm $\|\cdot\|$ of $\mathcal{X}$, with respect to which ψ is strongly convex. The simple FB descent method (using $\bar{x} = x^k$, $x^{k+1} = \hat{x}$) will then converge with essentially the same rate (but constants which depend on the new distance D_ψ); see Tseng (2008) for details. More interesting is the fact that, again, Tseng (2008) has also introduced accelerated variants which reach a convergence rate in $O(1/k^2)$, as before (see also Allen-Zhu and Orecchia 2014). A different way to introduce barriers and non-linearities for solving (4.26) by smoothing is proposed in Nesterov (2005), where another $O(1/k^2)$ algorithm is introduced.

[9] See also the version at http://www2.isye.gatech.edu/~nemirovs.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

The idea of considering non-linear proximity operator is not purely formal and can be useful. The most classical example is the case of optimization over the unit simplex

$$\left\{ x \in \mathbb{R}^n_+ : \sum_{i=1}^n x_i = 1 \right\}.$$

Then it is known (Teboulle 1992, Beck and Teboulle 2003) that the entropy

$$\psi(x) := \sum_{i=1}^n x_i \ln x_i \quad (\text{and } \nabla\psi(x) = (1 + \ln x_i)_{i=1}^n)$$

is 1-strongly convex with respect to the ℓ_1-norm[10]

$$\|x\|_1 = \sum_i |x_i|.$$

In this case, the (constrained) mirror step takes the form

$$\min_{\sum_i x_i = 1} \langle p, x\rangle + \frac{1}{\tau} D_\psi(x, \bar{x})$$

and the solution $\hat{x}$ satisfies

$$\ln \hat{x}_i = \ln \bar{x}_i - \tau p_i + \lambda,$$

where λ is a Lagrange multiplier for the constraint $\sum_i x_i = 1$. We obtain that, for $i = 1, \dots, n$,

$$\hat{x}_i = \frac{\mathrm{e}^{-\tau p_i}}{\sum_{j=1}^n \bar{x}_j \, \mathrm{e}^{-\tau p_j}} \bar{x}_i.$$

There might be two advantages: one is that we do not have to project back onto the simplex (although this projection is very cheap), the other is that the parameters of the problem in the ℓ_1-norm (such as the Lipschitz constant of the smooth part of the objective) might allow us to take a larger time step or yield better constants in the estimates for the rates (Beck and Teboulle 2003).

Non-linear smoothing or mirror descent is also useful for solving optimal transportation problems; for applications in imaging see Benamou *et al.* (2015) and Ferradans, Papadakis, Peyré and Aujol (2014).

4.8.2. Inertial and over-relaxed algorithms

The accelerated methods described in Theorems 4.5 and 4.10 are based on a particular example of 'overshooting', where the new point is obtained by

[10] This will imply that $D_\psi(x, x') \geq \|x - x'\|_1^2/2$, so even though $\|x\|_1 = \|x'\|_1 = 1$ it does carry some information!

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

applying an operator to the old point with a 'momentum' (here, a multiple of the difference between the two last iterates).

Gradient descent type methods can be accelerated in many similar ways. A very efficient method is the *heavy ball* (HB) method (Polyak 1987), which consists in iterating

$$x^{k+1} = x^k + \alpha\nabla f(x^k) + \beta(x^k - x^{k-1}). \tag{4.42}$$

For strongly convex problems, that is, assuming $\ell\,\mathrm{Id} \le \nabla^2 f \le L\,\mathrm{Id}$, this can be optimal: convergence is ensured for $0 \le \beta < 1$ and $0 < \alpha < 2(1+\beta)/L$, and the choices $\beta = q^2$, where

$$q = \frac{1-\sqrt{\ell/L}}{1+\sqrt{\ell/L}}, \quad \alpha = \frac{4}{(\sqrt{L}+\sqrt{\ell})^2}$$

yield the optimal rate $\|x^{k+1} - x^*\| = O(q^k)$ (Polyak 1987, Theorem 1).

The heavy ball method has been generalized to monotone operators by Alvarez and Attouch (2001) (see also Alvarez 2003, Moudafi and Oliny 2003), so there exist general convergence results that allow for non-smooth terms.

We should of course also mention the conjugate gradient descent method, which is of the same sort, except that the parameters α and β are updated dynamically at each iteration. Ideally we want to choose α, β which solve

$$\min_{\alpha,\beta} f(x^k + \alpha\nabla f(x^k) + \beta(x^k - x^{k-1}))$$

(see Polyak 1987). For a quadratic function this problem is easily solved, and it is known that the descent method obtained minimizes the quadratic function exactly in $\operatorname{rank} A$ iterations, where $A = \nabla^2 f$. It is the fastest method in this case (Polyak 1987); see the plot 'CG' in Figure 4.1. In practice, this method should be implemented on a sufficiently smooth problem when the cost of performing a line-search (which requires evaluations of the function) is not too large; as for non-quadratic problems, the optimal step cannot be computed in closed form.

A generalization of the HB algorithm to a strongly convex function given by the sum of a smooth, twice continuously differentiable function with Lipschitz-continuous gradient and a non-smooth function, with easily computed proximal map, was investigated for quadratic functions in Bioucas-Dias and Figueiredo (2007) and for more general smooth functions in Ochs, Brox and Pock (2015). It is of the form

$$x^{k+1} = \operatorname{prox}_{\alpha g}(x^k + \alpha\nabla f(x^k) + \beta(x^k - x^{k-1})). \tag{4.43}$$

The proximal HB algorithm offers the same optimal convergence rate as the HB algorithm, but can be applied only if the smooth function is twice continuously differentiable. It is therefore very efficient; see Figure 4.4 below for a comparison of this method with other accelerated methods.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Another standard and simple way to speed up such algorithms consists in simply over-relaxing the iterates, that is, replacing x^{k+1} with the value $x^{k+1} + \theta(x^{k+1} - x^k)$ such as in (4.8); this is not exactly the same as (4.30)–(4.31). Convergence is generally guaranteed as long as $\theta < 1$; this has been studied in a very general setting by Combettes and Wajs (2005). The theoretical convergence rates are in general only slightly improved by such over-relaxations, but sometimes the empirical rates are much better. On the other hand, there do not seem to be many studies of over-relaxed accelerated algorithms, although a recent paper on the 'FISTA' method shows that it is actually possible and improves the convergence (Yamagishi and Yamada 2011).

4.8.3. (Two) block(s) coordinate descent

It is obvious from the proof in Appendix B that any algorithm which ensures a descent rule such as (4.37) will enjoy the same convergence properties (Theorem 4.9) and can be accelerated by the same techniques as FB splitting. As a particular case, one can efficiently solve problems of the form

$$\min_{x\in\mathcal{X}} f_1(x) + f_2(x) + \frac{1}{2}\|x - x_0\|^2.$$

Indeed, in its dual formulation, this problem can be written as

$$\min_{y_1,y_2} f_1^*(y_1) + f_2^*(y_2) + \frac{1}{2}\|y_1 + y_2\|^2 - \langle y_1 + y_2, x_0\rangle,$$

and if we minimize successively with respect to y_1, y_2, it turns out that we obtain a descent rule similar to (4.37).

Lemma 4.13. Given $\bar{y}_1, \bar{y}_2$, let

$$\hat{y}_2 = \arg\min_{y_2} f_2^*(y_2) + \frac{1}{2}\|\bar{y}_1 + y_2\|^2 - \langle \bar{y}_1 + y_2, x_0\rangle,$$
$$\hat{y}_1 = \arg\min_{y_1} f_1^*(y_1) + \frac{1}{2}\|y_1 + \hat{y}_2\|^2 - \langle y_1 + \hat{y}_2, x_0\rangle.$$

Then, for all $(y_1, y_2) \in \mathcal{X}^2$, we have

$$f_1^*(y_1) + f_2^*(y_2) + \frac{1}{2}\|y_1 + y_2\| - \langle y_1 + y_2, x_0\rangle + \frac{1}{2}\|y_1 - \bar{y}_1\|^2$$
$$\geq f_1^*(\hat{y}_1) + f_2^*(\hat{y}_2) + \frac{1}{2}\|\hat{y}_1 + \hat{y}_2\|^2 - \langle \hat{y}_1 + \hat{y}_2, x_0\rangle + \frac{1}{2}\|y_1 - \hat{y}_1\|^2.$$

This is even improved if either f_1^* or f_2^* is strongly convex (equivalently, if at least one of the functions f_1 or f_2 has Lipschitz gradient). It clearly follows that the scheme of the proof of Theorem 4.10 will also work for this method: see Appendix B. The proof of the lemma is elementary. Moreover,

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

we can observe that since

$$\tilde{f}_2^* : y_1 \mapsto \min_{y_2} f_2^*(y_2) + \frac{1}{2}\|y_1 + y_2\|^2 - \langle y_1 + y_2, x_0 \rangle$$

is a convex function of y_1 with 1-Lipschitz gradient, the alternating minimization method is simply a forward–backward splitting applied to the problem $\min_{y_1} f_1^*(y_1) + \tilde{f}_2^*(y_1)$; see for instance Combettes and Pesquet (2011, Example 10.11). Less elementary is the fact that this descent rule still holds if the exact minimizations are replaced with proximal (implicit descent) steps or if the quadratic part is linearized, which can be useful if it involves linear operators; see Chambolle and Pock (2015*b*) for details.

A particular case of this splitting is used in Section 7.8 to compute Figure 7.9; see the explanations there (Chambolle and Pock 2015*b*, Kolmogorov, Pock and Rolinek 2016). It can also be used to implement fast parallel solvers for the ROF problem (2.6): the idea is to split the dual variable into two groups, one 'living' on the 'odd' squares (or cubes in three dimensions), that is, the edges connecting the vertices $(i, j) + \{0, 1\}^2$, i, j odd, and the other in the 'even' squares. Then one can use a dedicated solver to solve exactly or approximately the subproblem on each odd/even square, which are low-dimensional decoupled problems. This is particularly well adapted to implementation on GPUs; details can be found in Chambolle and Pock (2015*b*).

In general, block coordinate (or Gauss–Seidel) descent schemes can be implemented in many ways, and many generalizations involving non-smooth terms are proved to converge in the literature (Grippo and Sciandrone 2000, Auslender 1976, Attouch *et al.* 2013). As long as some energy decay is guaranteed, $O(1/k)$ rates are easy to prove. In the context of this chapter, see in particular Beck and Tetruashvili (2013); see also Tseng (2001), Tseng and Yun (2009), Beck (2015), Chouzenoux, Pesquet and Repetti (2016) and Nesterov (2012).

For more than two blocks, efficient methods can be developed in two different directions: sums of 'simple' objective can be dualized and their proximity operators then computed in parallel (Attouch, Briceño-Arias and Combettes 2009/10, Raguet, Fadili and Peyré 2013, Becker and Combettes 2014, Pustelnik, Chaux and Pesquet 2011). Acceleration is then possible in this framework (Goldfarb and Ma 2012). On the other hand, randomized algorithms seem to be a very efficient alternative for tackling problems with a huge number of variables or blocks (Nesterov 2012). In particular, whereas in the deterministic setting it is hard to implement acceleration techniques for problems involving more than two blocks, stochastic block descent methods will typically average out antisymmetric terms in the descent rules and lead to much nicer inequalities which can be exploited to derive very efficient methods (Lin, Lu and Xiao 2015). A few recent

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

algorithms recover optimal rates (in particular when specialized to the one-block case) and allow for descent steps which are optimal for each block (Fercoq and Richtárik 2015, 2013).

4.8.4. FBF splitting

In the context of maximal monotone operators, an important generalization of the FB splitting algorithm is due to Tseng (2000). The standard FB splitting algorithm requires the forward operator to be co-coercive, for example the gradient of a smooth function. This clearly limits the applicability of the algorithm to more general problems. The following modification, called the *forward–backward–forward* (FBF) algorithm simply assumes that the forward monotone operator is single-valued and Lipschitz-continuous. It can therefore be applied, for example, if the forward operator is a skew-symmetric matrix. Let A, B be two maximal monotone operators with A single-valued on $\mathrm{dom}A \supset \mathrm{dom}B$. The FBF algorithm is defined by the following scheme:

$$x^{k+1/2} = (I + \tau_k B)^{-1}(I - \tau_k A)(x^k), \tag{4.44}$$

$$x^{k+1} = \Pi_{\mathcal{X}}\big(x^{k+1/2} - \tau_k(A(x^{k+1/2}) - A(x^k))\big), \tag{4.45}$$

where $\mathcal{X}$ is a suitable non-empty set (*e.g.* $\mathbb{R}^n$) and τ_k is the largest number satisfying, for any $\delta \in (0,1)$,

$$\tau_k \|A(x^{k+1/2}) - A(x^k)\| \leq \delta \|x^{k+1/2} - x^k\|,$$

which in practice can be determined by an Armijo-type backtracking procedure (Armijo 1966). An important application of this algorithm is to convex-concave saddle-point problems, which we will investigate in more detail in the next section.

4.9. Examples

We conclude this section by providing two examples. In the first example we consider minimizing the dual of the Huber-ROF problem, which is strongly convex and can therefore be minimized using accelerated proximal gradient descent for strongly convex problems. The second example uses the explicit representation of Moreau–Yosida regularization to transform the dual of an anisotropic variant of the ROF model into a form consisting of a smooth plus a non-smooth function, which can be tackled by accelerated forward–backward splitting.

Example 4.14 (minimizing the dual of Huber-ROF). Let us revisit the dual of the Huber-ROF model introduced in (4.21):

$$\min_{\mathbf{p}} \frac{1}{2}\|\mathrm{D}^*\mathbf{p} - u^\diamond\|^2 + \frac{\varepsilon}{2\lambda}\|\mathbf{p}\|^2 + \delta_{\{\|\cdot\|_{2,\infty} \leq \lambda\}}(\mathbf{p}),$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

where $u^\diamond$ is again the noisy image of size $m \times n$ from Example 2.1, and D is the (two-dimensional) finite difference operator. This problem is the sum of a smooth function with Lipschitz-continuous gradient,

$$f(\mathbf{p}) = \frac{1}{2}\|\mathrm{D}^*\mathbf{p} - u^\diamond\|^2,$$

plus a non-smooth function with easily computed proximal map,

$$g(\mathbf{p}) = \frac{\varepsilon}{2\lambda}\|\mathbf{p}\|^2 + \delta_{\{\|\cdot\|_{2,\infty}\le\lambda\}}(\mathbf{p}).$$

The gradient of the smooth function is given by

$$\nabla f(\mathbf{p}) = \mathrm{D}(\mathrm{D}^*\mathbf{p} - u^\diamond),$$

and its Lipschitz parameter is estimated again as $L \le 8$. The non-smooth function is strongly convex with parameter $\mu = \varepsilon/\lambda$ and its pixelwise proximal map is given by

$$\hat{\mathbf{p}} = \mathrm{prox}_{\tau g}(\tilde{\mathbf{p}}) \Leftrightarrow \hat{\mathbf{p}}_{i,j} = \frac{(1+\tau\mu)^{-1}\,\tilde{\mathbf{p}}_{i,j}}{\max\{1, (1+\tau\mu)^{-1}\,|\tilde{\mathbf{p}}_{i,j}|_2\}}$$

Let us now apply the Huber-ROF model to the image in Example 2.1 using the parameters $\lambda = 0.1$ and $\varepsilon = 0.001$. We implemented the FISTA algorithm (Algorithm 5) using the extrapolation parameters corresponding to $\mu = 0$ and the correct $\mu = \varepsilon/\lambda$. For comparison, we also implemented the proximal heavy ball algorithm (4.43) and used the optimal parameter settings

$$\alpha = \frac{4}{(\sqrt{\mu} + \sqrt{L+\mu})^2 - 4\mu}, \quad \beta = \frac{(\sqrt{\mu} - \sqrt{L+\mu})^2}{(\sqrt{\mu} + \sqrt{L+\mu})^2 - 4\mu}.$$

Figure 4.4 shows that it is generally not a good idea to apply the classical FISTA algorithm using $\mu = 0$ to a strongly convex problem. On the other hand, applying the FISTA algorithm with the correct settings for the strong convexity, that is, $\mu = \varepsilon/\lambda$, largely improves the convergence rate of the algorithm. Interestingly, it turns out that the proximal HB algorithm converges almost twice as fast as the FISTA algorithm (ω^k as opposed to ω^{2k} with $q = L/\mu_g$ and $\omega = (\sqrt{q}-1)/(\sqrt{q}+1)$). In fact the proximal HB algorithm seems to exactly obey the lower bound of first-order algorithms for the strongly convex problems presented in Theorem 4.14.

Example 4.15 (total variation on chains). We have already seen that when the smooth function is quadratic, the forward–backward algorithm is equivalent to a plain gradient method applied to Moreau–Yosida regularization. The aim of this example is to give a practical problem where such

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

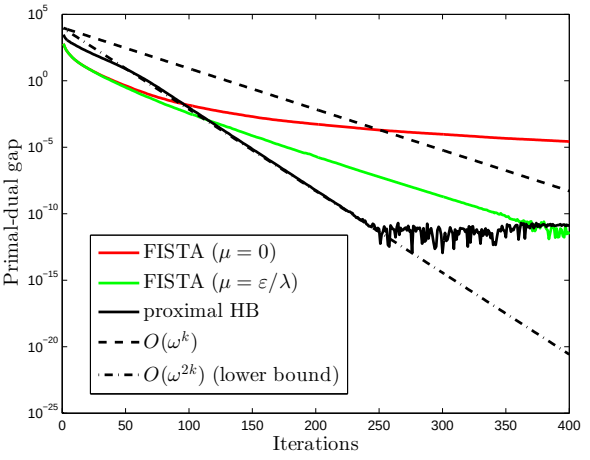


Figure 4.4. Convergence of accelerated proximal gradient descent methods for minimizing the dual Huber-ROF model using the image in Figure 2.1. Using the correct modulus of strong convexity ($\mu = \varepsilon/\lambda$), the FISTA algorithm performs much better than the FISTA algorithm, which does not take into account the correct value of μ. Interestingly, a tuned proximal heavy ball (HB) algorithm that uses the correct value of μ clearly outperforms FISTA and seems to coincide with the lower bound of first-order methods.

an equivalence does not hold. Consider again the dual of the ROF model:

$$\min_{\mathbf{p}} \frac{1}{2}\|\mathrm{D}^*\mathbf{p} - u^\diamond\|^2 + \delta_{\{\|\cdot\|_\infty \le \lambda\}}(\mathbf{p}), \tag{4.46}$$

which differs slightly from our previous ROF problems by the choice of the norm constraining the dual variables. First, application of the adjoint of the finite difference operator to the dual variables $\mathbf{p} = (p_1, p_2)$ can be decomposed via

$$\mathrm{D}^*\mathbf{p} = \sum_{d=1}^{2} \mathrm{D}_d^* p_d,$$

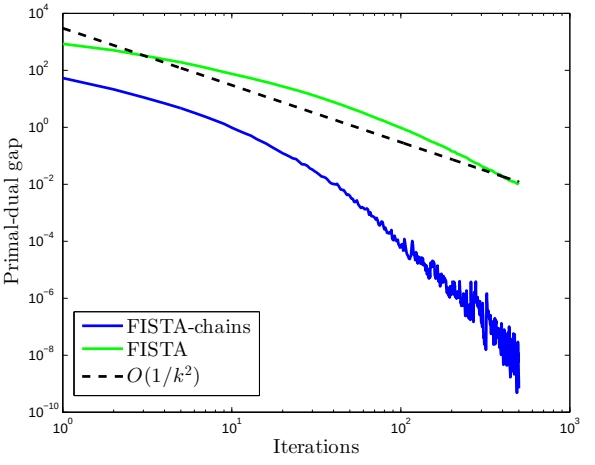
where D_d^* is the adjoint finite difference operator in the direction d. Second, by a change of variables $t_d = \mathrm{D}_d^* p_d$ and using the property that the constraint on $\mathbf{p}$ is also decomposable, we can rewrite the problem in the equivalent form

$$\min_{(t_d)_{d=1}^2} \frac{1}{2}\|\sum_{d=1}^{2} t_d - u^\diamond\|^2 + \sum_{d=1}^{2} \delta_{C_d}(t_d), \tag{4.47}$$

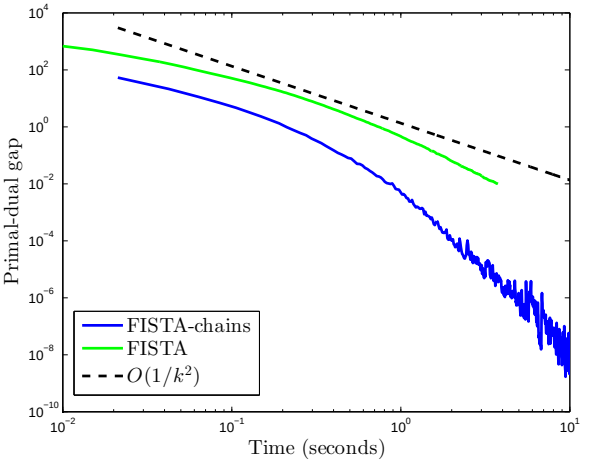
where

$$C_d = \{t_d : t_d = \mathrm{D}_d^* p_d, \|p_d\|_\infty \le \lambda\}, \quad \text{for } d = 1, 2.$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) iterations



(b) CPU time

Figure 4.5. Minimizing the dual ROF model applied to the image in Figure 2.1. This experiment shows that an accelerated proximal block descent algorithm (FISTA-chains) that exactly solves the ROF problem on horizontal and vertical chains significantly outperforms a standard accelerated proximal gradient descent (FISTA) implementation. (a) Comparison based on iterations, (b) comparison based on the CPU time.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Hence, as shown in Section 4.8.3, this problem could be easily solved via accelerated alternating minimization in t_d if we were able to efficiently compute the proximal maps with respect to $\delta_{C_d}(t_d)$. Moreover, we have shown that the (accelerated) alternating minimization corresponds to an (accelerated) forward–backward algorithm on the partial Moreau–Yosida regularization that is obtained by partially minimizing (4.47) with respect to one variable, hence corresponding to a non-trivial instance of the forward–backward algorithm. Observe that the characteristic functions of the sets C_d are exactly the convex conjugates of the total variation in each dimension d, that is,

$$\delta_{C_d}(t_d) = \sup_u \langle u, t_d \rangle - \lambda \|\mathrm{D}_d u\|_1.$$

In other words, if we were able to solve the proximal maps for one-dimensional total variation problems along chains, we could – thanks to Moreau's identity – also efficiently solve the proximal maps for the functions $\delta_{C_d}(t_d)$.

As a matter of fact, there exist several direct algorithms that can solve one-dimensional ROF problems very efficiently, and hence the proximal maps for one-dimensional total variation. Some of the algorithms even work in linear time; see Davies and Kovac (2001), Condat (2013*a*), Johnson (2013) and Kolmogorov *et al.* (2016), and references therein.

Figure 4.5 presents a comparison between the convergence rates of accelerated block descent (FISTA-chains) applied to (4.47) and a standard implementation of FISTA applied to (4.46). To solve the one-dimensional total variation subproblems on chains we used the linear-time dynamic programming approach from Kolmogorov *et al.* (2016). Figure 4.5(a) shows that in terms of iterations, the accelerated block descent is about 10–20 times as fast. Clearly, one iteration of the accelerated block descent is computationally more expensive compared to one iteration of the standard implementation; in our C++ implementation, one iteration of standard FISTA was approximately three times as fast compared to the accelerated block descent. Yet overall the block splitting technique turns out to be more efficient for a given precision, as shown in Figure 4.5(b). Later, in Section 7.8, we will come back to a similar example and show how accelerated block descent can be used to solve large-scale stereo problems.

5. Saddle-point methods

In this section we will briefly describe the main optimization techniques for finding saddle points, which are commonly used for imaging problems. The goal of these approaches is, as before, to split a complex problem into simpler subproblems which are easy to solve – although depending on the structure and properties of the functions, one form might be more suitable than another. We will mostly concentrate on one type of algorithm known as

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

the 'primal–dual' algorithm, 'ADMM', or 'Douglas–Rachford splitting' (see references below) in a Euclidean setting, although more complex splitting techniques can be useful (*e.g.* Tseng 2000), as well as descent with respect to non-linear metrics or in Banach spaces (Nemirovski 2004, Chambolle and Pock 2015*a*). We will mention the simplest useful results. These have been generalized and improved in many ways; see in particular Davis (2015) and Davis and Yin (2014*a*, 2014*b*) for an extensive study of convergence rates, Chen, Lan and Ouyang (2014*a*), Ouyang, Chen, Lan and Pasiliao (2015) and Valkonen and Pock (2015) for optimal methods exploiting partial regularity of some objectives, and Fercoq and Bianchi (2015) for efficient stochastic approaches.

The natural order in which to present these algorithms should be to start with the Douglas–Rachford splitting (Douglas and Rachford 1956; the modern form we will describe is found in Lions and Mercier 1979) and the ADMM, which have been used for a long time in non-smooth optimization. However, since the convergence results for primal–dual methods are in some sense much simpler and carry on to the other algorithms, we first start by describing these methods.

5.1. Primal–dual algorithms

The problems we consider here are in the 'standard' form (3.9)

$$\min_{x\in\mathcal{X}} f(Kx) + g(x),$$

where f, g are convex, l.s.c. and 'simple', and $K : \mathcal{X} \to \mathcal{Y}$ is a bounded linear operator. When f is smooth, FB splitting can be used efficiently for such a problem. In other situations we usually have to revert to Lagrangian techniques or primal–dual methods. This is the case for Examples 2.2 and 2.3 below, for example.

The idea is to write the problem as a saddle point as in (3.10):

$$\max_y \inf_x \langle y, Kx\rangle - f^*(y) + g(x).$$

Then (this dates back to Arrow, Hurwicz and Uzawa 1958), we alternate a (proximal) descent in the variable x and an ascent in the dual variable y:

$$x^{k+1} = \operatorname{prox}_{\tau g}(x^k - \tau K^* y^k), \tag{5.1}$$

$$y^{k+1} = \operatorname{prox}_{\sigma f^*}(y^k + \sigma K x^{k+1}). \tag{5.2}$$

It is not clear that such iterations will converge. (We can easily convince ourselves that a totally explicit iteration, with x^{k+1} above replaced with x^k, will in general not converge.) However, this scheme was proposed in Zhu and Chan (2008) for problem (2.6) and observed to be very efficient for this problem, especially when combined with an acceleration strategy

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 6 PDHG.

Input: initial pair of primal and dual points (x^0, y^0), steps $\tau, \sigma > 0$.
for all $k \geq 0$ **do**
find (x^{k+1}, y^{k+1}) by solving

$$x^{k+1} = \mathrm{prox}_{\tau g}(x^k - \tau K^* y^k) \tag{5.3}$$

$$y^{k+1} = \mathrm{prox}_{\sigma f^*}(y^k + \sigma K(2x^{k+1} - x^k)). \tag{5.4}$$

end for

consisting in decreasing τ and increasing σ at each step (*e.g.*, following the rules in Algorithm 8 below). Proofs of convergence for the Zhu–Chan method have been proposed by Esser, Zhang and Chan (2010), Bonettini and Ruggiero (2012) and He, You and Yuan (2014). For a general problem there exist several strategies to modify these iterations into converging subsequences. Popov (1981) proposed incorporating a type of 'extragradient' strategy into these iterations, as introduced by Korpelevich (1976, 1983): the idea is simply to replace y^k with $\mathrm{prox}_{\sigma f^*}(y^k + \sigma K^* x^k)$ in (5.1). This makes the algorithm convergent; moreover, an $O(1/k)$ (ergodic) convergence rate is shown in Nemirovski (2004) (for a class of schemes including this one, using also non-linear 'mirror' descent steps: see Section 4.8.1). A variant with similar properties, but not requiring us to compute an additional step at each iteration, was proposed at roughly the same time by Esser *et al.* (2010) (who gave it the name 'PDHG'[11]), and Pock, Cremers, Bischof and Chambolle (2009). The iterations can be written as in Algorithm 6.

The over-relaxation step $2x^{k+1} - x^k = x^{k+1} + (x^{k+1} - x^k)$ can be interpreted as an approximate extragradient, and indeed it is possible to show convergence of this method with a rate which is the same as in Nemirovski (2004) (see also Chambolle and Pock 2011, 2015*a*). On the other hand, this formula might recall similar relaxations present in other standard splitting algorithms such as the Douglas–Rachford splitting or the ADMM (see Sections 5.3 and 5.4 below), and indeed, we then see that this algorithm is merely a variant of these other methods, in a possibly degenerate metric. He *et al.* (2014) observed that, letting $z = (x, y)$, the iterations above can be written as

$$M(z^{k+1} - z^k) + Tz^{k+1} \ni 0, \tag{5.5}$$

[11] *Primal–dual hybrid gradient.* More precisely, the algorithm we describe here would correspond to 'PDHGMu' and 'PDHGMp' in Esser *et al.* (2010), while 'PDHG' correspond to a plain Arrow–Hurwicz alternating scheme such as in Zhu and Chan (2008). However, for simplicity we will keep the name 'PDHG' for the general converging primal–dual method.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 7 General form of primal–dual iteration.

Input: previous points $(\bar{x}, \bar{y}, \tilde{x}, \tilde{y})$, steps $\tau, \sigma > 0$.
Output: new points $(\hat{x}, \hat{y}) = \mathcal{PD}_{\tau,\sigma}(\bar{x}, \bar{y}, \tilde{x}, \tilde{y})$ given by

$$\begin{cases} \hat{x} = \operatorname{prox}_{\tau g}(\bar{x} - \tau(\nabla h(\bar{x}) + K^* \tilde{y})), \\ \hat{y} = \operatorname{prox}_{\sigma f^*}(\bar{y} + \sigma K \tilde{x}). \end{cases} \tag{5.6}$$

where T is the monotone operator in (3.15) and M is the metric

$$M = \begin{pmatrix} \frac{1}{\tau} I & -K^* \\ -K & \frac{1}{\sigma} I \end{pmatrix}, \tag{5.7}$$

which is positive definite if $\tau\sigma\|K\|^2 < 1$. Hence, in this form the primal–dual algorithm is simply a proximal-point algorithm applied to the monotone operator T, and standard convergence results or rates (Brézis and Lions 1978) can be deduced.

This can be extended to a slightly more general form. Assume that we want to minimize the problem

$$\min_{x \in \mathcal{X}} f(Kx) + g(x) + h(x), \tag{5.8}$$

where f, g are convex, l.s.c. and 'simple', K is a bounded linear operator and h is convex with L_h-Lipschitz gradient term which we will treat explicitly. The primal–dual method for such a problem was suggested by Condat (2013*b*) and its convergence studied by Vũ (2013*a*) and Boţ, Csetnek, Heinrich and Hendrich (2015) (the latter papers dealing with more general monotone operators). Rates of convergence, including control of the primal–dual gap, are established in Chambolle and Pock (2015*a*) (a variant is studied in Drori, Sabach and Teboulle 2015), and a close (different) algorithm which mixes general monotone operators and subgradients and establishes similar rates is found in Davis and Yin (2015). The idea is simply to replace the descent step in x with an FB splitting step, letting

$$x^{k+1} = \operatorname{prox}_{\tau g}(x^k - (\tau K^* y^k + \tau \nabla h(x^k)).$$

Let us now write the algorithm in a general form, as in Algorithm 7. The first case (5.3)–(5.4) corresponds to iterations with fixed steps τ, σ, $\bar{x} = x^k$, $\bar{y} = \tilde{y} = y^k$, $\tilde{x} = 2x^{k+1} - x^k$. In this case we have the following result, which also obviously applies to the PDHG method (5.3)–(5.4). Here we let $L = \|K\|$.

Theorem 5.1. Let $\tau, \sigma > 0$ and $(x^0, y^0) \in \mathcal{X} \times \mathcal{Y}$ be given, and for $k \geq 0$ let

$$(x^{k+1}, y^{k+1}) = \mathcal{PD}_{\tau,\sigma}(x^k, y^k, 2x^{k+1} - x^k, y^k).$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Assume

$$\left(\frac{1}{\tau} - L_h\right)\frac{1}{\sigma} \geq L^2. \tag{5.9}$$

Then, for any $(x, y) \in \mathcal{X} \times \mathcal{Y}$, we have

$$\mathcal{L}(X^k, y) - \mathcal{L}(x, Y^k) \leq \frac{\frac{1}{\tau}\|x - x^0\|^2 + \frac{1}{\sigma}\|y - y^0\|^2}{k}, \tag{5.10}$$

where[12]

$$X^k = \frac{1}{k}\sum_{i=1}^{k} x^i, \quad Y^k = \frac{1}{k}\sum_{i=1}^{k} y^i.$$

Moreover, if the inequality is strict in (5.9), then (x^k, y^k) converge (weakly in infinite dimension) to a saddle point.

Proof. This is a particular case of Theorem 1, Remark 1 and Remark 3 in Chambolle and Pock (2015*a*). Under additional assumptions, one can derive a similar rate of convergence for the 'true' primal–dual gap $\mathcal{G}(X^k, Y^k)$. □

Note that this result says little, in general, about the 'best' choice of the parameters τ, σ, and the empirical speed of convergence often depends a lot on this choice. Heuristic approaches have been proposed (which in general try to ensure that the primal and dual variables evolve at roughly the 'same speed'); an efficient one, together with a backtracking strategy and convergence guarantees, is proposed in Goldstein *et al.* (2015).

Acceleration

An interesting feature of these types of primal–dual iteration is the fact they can be 'accelerated', in cases when the objective function has more regularity. The first case is when $g + h$ (or f^*) is strongly convex: see Algorithm 8. Observe that if f^* is μ_f-strongly convex, then $x \mapsto f(Kx)$ has (L^2/μ_f)-Lipschitz gradient, and it is natural to expect that one will be able to decrease the objective at rate $O(1/k^2)$ as before. Similarly, we expect the same if g or h is strongly convex. This is the result we now state. We should assume here that g is μ_g-convex, h is μ_h-convex, and $\mu = \mu_g + \mu_h > 0$. However, in this case it is no different to assuming that g is μ-convex, as one can always replace h with $h(x) - \mu_h\|x\|^2/2$ (which is convex with $(L_h - \mu_h)$-Lipschitz gradient $\nabla h(x) - \mu_h x$), and g with $g(x) + \mu_h\|x\|^2/2$ (whose proximity operator is as easy to compute as g's). For notational simplicity, we will thus restrict ourselves to this latter case – which is equivalent to the general case upon replacing τ with $\tau' = \tau/(1 + \tau\mu_h)$.

[12] This is called an 'ergodic' convergence rate.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 8 Accelerated primal–dual algorithm 1.

Choose $\tau_0 = 1/(2L_h)$ and $\sigma_0 = L_h/L^2$ (or any τ_0, σ_0 with $\tau_0\sigma_0 L^2 \leq 1$ if $L_h = 0$), $\theta_0 = 0$ and $x^{-1} = x^0 \in \mathcal{X}$, $y^0 \in \mathcal{Y}$,
for all $k \geq 0$ **do**
$(x^{k+1}, y^{k+1}) = \mathcal{PD}_{\tau_k,\sigma_k}(x^k, y^k, x^k + \theta_k(x^k - x^{k-1}), y^{k+1})$,
$\theta_{k+1} = 1/\sqrt{1 + \mu_g\tau_k}$, $\tau_{k+1} = \theta_{k+1}\tau_k$, $\sigma_{k+1} = \sigma_k/\theta_{k+1}$.
end for

Theorem 5.2. Let $(x^k, y^k)_{k\geq 0}$ be the iterations of Algorithm 8. For each $k \geq 1$, define $t_k = \sigma_{k-1}/\sigma_0$, $T_k = \sum_{i=1}^k t_i$ and the averaged points

$$(X^k, Y^k) = \frac{1}{T_k}\sum_{i=1}^k t_i(x^i, y^i).$$

Then for any $k \geq 1$ and any $(x, y) \in \mathcal{X} \times \mathcal{Y}$,

$$T_k\big(\mathcal{L}(X^k, y) - \mathcal{L}(x, Y^k)\big) + \frac{t_{k+1}^2}{2\tau_0}\|x^k - x\|^2 \leq \frac{1}{2\tau_0}\|x^0 - x\|^2 + \frac{1}{2\sigma_0}\|y^0 - y\|^2. \tag{5.11}$$

One can then show that with this choice $t_k \approx \gamma k/(4L_f)$, so that also $1/T_k = O(1/k^2)$ (see Chambolle and Pock 2011). Under additional assumptions (for instance if f has full domain, so that f^* is superlinear), it follows a global $O(1/k^2)$ estimate for the primal–dual gap $\mathcal{G}(X^k, Y^k)$, although with a constant which could be very large.

Proof. This is a particular case of Chambolle and Pock (2015*a*, Theorem 4). □

Remark 5.3. We should mention here that the over-relaxation step above can also be performed in the y variable, therefore letting

$$(x^{k+1}, y^{k+1}) = \mathcal{PD}_{\tau_k,\sigma_k}(x^k, y^k, x^{k+1}, y^k + \theta_k(y^k - y^{k-1})).$$

The proof remains identical, but since it is not widely known we sketch it in Appendix C.2. It may seem to be quite a trivial remark, but in Section 5.3 we will see that it can be useful.

Now let us assume that f^* is also strongly convex with parameter $\mu_{f^*} = 1/L_f$. Then an appropriate choice of the parameter σ, τ, θ can yield a better (linear) convergence rate: see Algorithm 9. We give a particular choice of parameters for which such convergence occurs, but note that it is possible to show linear convergence with quite general over-relaxation parameters (but

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 9 Accelerated primal–dual algorithm 2.

Choose $x^{-1} = x^0 \in \mathcal{X}$, $y^0 \in \mathcal{Y}$, and $\tau, \sigma, \theta > 0$ satisfying $\theta^{-1} = 1 + \mu_g \tau = 1 + \mu_{f^*}\sigma$ and $\theta L^2 \sigma\tau \leq 1 - L_h\tau$.
for all $k \geq 0$ **do**
$(x^{k+1}, y^{k+1}) = \mathcal{PD}_{\tau,\sigma}(x^k, y^k, x^k + \theta(x^k - x^{k-1}), y^{k+1})$,
end for

the global rate is strongly dependent on the parameter choice: for details see Chambolle and Pock 2011 or Tan 2016). Letting

$$\alpha = \frac{\mu_{f^*}(\mu_g + L_h)}{2L^2}\left(\sqrt{1 + 4\frac{\mu_g L^2}{\mu_{f^*}(\mu_g + L_h)^2}} - 1\right) \in (0,1),$$

a possible choice for τ, σ, θ is given by

$$\tau = \frac{\alpha}{\mu_g(1-\alpha)}, \quad \sigma = \frac{\alpha}{\mu_{f^*}(1-\alpha)}, \quad \theta = 1 - \alpha \in (0,1).$$

We can then show the following ergodic estimate.

Theorem 5.4. Let $(x^k, y^k)_{k\geq 0}$ be the iterations of Algorithm 9. For each $k \geq 1$, define $t_k = \sigma_{k-1}/\sigma_0$, $T_k = \sum_{i=1}^k \theta^{-i+1}$ and the averaged points

$$(X^k, Y^k) = \frac{1}{T_k}\sum_{i=1}^k \theta^{-i+1}(x^i, y^i).$$

Then, for any $k \geq 1$ and any $(x, y) \in \mathcal{X} \times \mathcal{Y}$,

$$\mathcal{L}(X^k, y) - \mathcal{L}(x, Y^k) \leq \frac{1}{T_k}\left(\frac{1}{2\tau}\|x^0 - x\|^2 + \frac{1}{2\sigma}\|y^0 - y\|^2\right). \tag{5.12}$$

Observe that $1/T_k = O(\theta^k)$, so this is indeed a linear convergence rate.

Proof. See Chambolle and Pock (2015*a*, Theorem 5). □

Preconditioning

As a quick remark, we mention here that it is not always obvious how to estimate the norm of the matrix $L = \|K\|$ precisely and efficiently, without which we cannot choose parameters correctly. An interesting use of general preconditioners is suggested by Bredies and Sun (2015*a*, 2015*b*) for the variants of the algorithm described in the next few sections. The main difficulty is that if the metric is changed, f and g might no longer be 'simple'. A simpler approach is suggested in Pock and Chambolle (2011), for problems where a diagonal preconditioning does not alter the property that the proximal operators of f and g are easy to compute. Let us briefly describe

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

a variant which is very simple and allows for a large choice of diagonal preconditioners. If we assume $h = 0$, then the PDHG algorithm of Theorem 5.1 can be written equivalently as a proximal-point iteration such as (5.5) (He *et al.* 2014). Changing the metric means replacing M in (5.7) with

$$M' = \begin{pmatrix} T^{-1} & -K^* \\ -K & \Sigma^{-1} \end{pmatrix},$$

where T and Σ are positive definite symmetric matrices. This means that the prox operators in iteration (5.6) must be computed in the new metrics T^{-1} and Σ^{-1}: in other words, the points $(\hat{x}, \hat{y})$ are replaced with the solutions of

$$\min_x \frac{1}{2}\|x - \bar{x}\|^2_{T^{-1}} + \langle \tilde{y}, Kx\rangle + g(x), \quad \min_y \frac{1}{2}\|y - \bar{y}\|^2_{\Sigma^{-1}} - \langle y, K\tilde{x}\rangle + f^*(y).$$

Using a diagonal preconditioning means that we also require T and Σ to be diagonal. The reason to impose this is that in many cases the proximity operators of f and g will remain simple in such metrics, and might become intractable in more general metrics. We should mention here that Becker and Fadili (2012) have shown that, in problems with separable simple functions, one can in fact use preconditioners that are diagonal+rank one, without altering the simplicity of the functions. These authors use this remark to build quasi-Newton descent schemes, but it might also be interesting in a primal–dual framework.

A necessary condition for the algorithm to be well posed (which is then sufficient for convergence, as shown in Theorem 5.1), is that M' is positive (semi-)definite. An elementary computation shows that it is equivalent to the requirement

$$\|\Sigma^{1/2} K T^{1/2}\| \leq 1. \tag{5.13}$$

The following strategy, which extends the choice in Pock and Chambolle (2011), allows us to design matrices Σ and T such that this holds. We assume here that $\mathcal{X} = \mathbb{R}^n$ and $\mathcal{Y} = \mathbb{R}^m$, $m, n \geq 1$, so K is an $(m \times n)$-matrix.

Lemma 5.5. Let $(\tilde{\tau}_i)_{1\leq i\leq n}$ and $(\tilde{\sigma}_j)_{1\leq j\leq m}$ be arbitrary positive numbers, and $\alpha \in [0, 2]$. Then let

$$T = \mathrm{diag}((\tau_i)_{i=1}^n), \quad \Sigma = \mathrm{diag}((\sigma_j)_{j=1}^m),$$

where

$$\tau_i = \frac{\tilde{\tau}_i}{\sum_{j=1}^m \tilde{\sigma}_j |K_{j,i}|^{2-\alpha}}, \quad \sigma_j = \frac{\tilde{\sigma}_j}{\sum_{i=1}^n \tilde{\tau}_i |K_{j,i}|^{\alpha}}.$$

Then (5.13) holds.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Proof. The proof is as in Pock and Chambolle (2011): for any $x \in X$, we just observe that

$$\begin{aligned}\sum_{j=1}^{m}\Big(\sum_{i=1}^{n}\sqrt{\sigma_j}K_{j,i}\sqrt{\tau_i}x_i\Big)^2 &\le \sum_{j=1}^{m}\sigma_j\Big(\sum_{i=1}^{n}|K_{j,i}|^{\alpha/2}\sqrt{\tilde\tau_i}\frac{|K_{j,i}|^{1-\alpha/2}}{\sqrt{\tilde\tau_i}}\sqrt{\tau_i}|x_i|\Big)^2\\ &\le \sum_{j=1}^{m}\sigma_j\Big(\sum_{i=1}^{n}|K_{j,i}|^{\alpha}\tilde\tau_i\Big)\Big(\sum_{i=1}^{n}\frac{|K_{j,i}|^{2-\alpha}}{\tilde\tau_i}\tau_i x_i^2\Big)\\ &= \sum_{j=1}^{m}\tilde\sigma_j\Big(\sum_{i=1}^{n}\frac{|K_{j,i}|^{2-\alpha}}{\sum_{l=1}^{m}\tilde\sigma_l|K_{l,i}|^{2-\alpha}}x_i^2\Big) = \sum_{i=1}^{n}x_i^2, \end{aligned} \tag{5.14}$$

which shows (5.13). □

It means that with this choice the algorithm will converge. This can be extended to accelerated variants if, for instance, g is strongly convex, provided that T is chosen in such a way that the strong convexity parameter in the new metric does not degenerate too much (in particular, such preconditioning should be useful if g is strongly convex but with a very small parameter in some variables). The case $\alpha = 0$ was suggested by Eckstein (1989) for primal–dual iterations, referred to as the 'alternating step method for monotropic programming'. This is simply the PDHG algorithm applied to particular instances of (3.9), with a simpler structure.

We now show that the PDHG algorithm and its variants can be used to efficiently minimize our three introductory examples provided in Section 2.2.

Example 5.6 (ROF model using accelerated PDHG). In this example, we show how to implement the PDHG algorithm to minimize the ROF model (2.6) used in Example 2.1, which we recall here for convenience:

$$\min_u \lambda\|\mathrm{D}u\|_{2,1} + \frac{1}{2}\|u - u^\diamond\|^2,$$

where $u^\diamond$ is the noisy input image. First we need to transform the objective function to a form that can be tackled by the PDHG algorithm. Using duality, we obtain the saddle-point problem

$$\min_u \max_{\mathbf{p}} \langle \mathrm{D}u, \mathbf{p}\rangle + \frac{1}{2}\|u - u^\diamond\|^2 - \delta_{\{\|\cdot\|_{2,\infty}\le\lambda\}}(\mathbf{p}).$$

It remains to detail the implementation of the proximity operators for the functions $g(u) = \frac{1}{2}\|u - u^\diamond\|^2$ and $f^*(\mathbf{p}) = \delta_{\{\|\cdot\|_{2,\infty}\le\lambda\}}(\mathbf{p})$. The proximal map for the function $g(u)$ is given by solving a simple quadratic problem for each

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

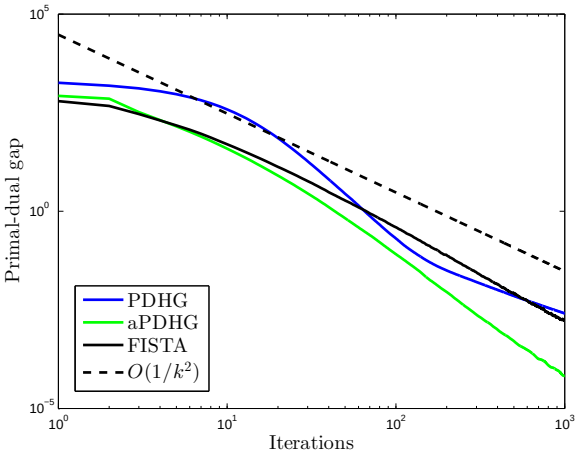


Figure 5.1. Minimizing the ROF model applied to the image in Figure 2.1. This experiment shows that the accelerated primal–dual method with optimal dynamic step sizes (aPDHG) is significantly faster than a primal–dual algorithm that uses fixed step sizes (PDGH). For comparison we also show the performance of accelerated proximal gradient descent (FISTA).

pixel. For a given $\tilde{u}$, the proximal map is given by

$$\hat{u} = \text{prox}_{\tau g}(\tilde{u}) \Leftrightarrow \hat{u}_{i,j} = \frac{\tilde{u}_{i,j} + \tau u^{\diamond}_{i,j}}{1+\tau}.$$

The proximal map for the function $f^*(\mathbf{p})$ is given by the pixelwise orthogonal projection onto ℓ_2-balls with radius λ. This projection can be easily computed using formula (4.23).

We implemented both the standard PDHG algorithm (Algorithm 6) and its accelerated variant (Algorithm 8) and applied it to the image from Example 2.1. For comparison we also ran the FISTA algorithm (Algorithm 5) on the dual ROF problem. For the plain PDHG we used a fixed setting of the step sizes $\tau = 0.1$, $\sigma = 1/(\tau L^2)$, where $L = \|\mathrm{D}\| \leq \sqrt{8}$. For the accelerated PDHG (aPDHG), we observe that the function $g(u)$ is $(\mu_g = 1)$-strongly convex, and we used the proposed settings for dynamically updating the step size parameters. The initial step size parameters were set to $\tau_0 = \sigma_0 = 1/L$.

Figure 5.1 shows the decay of the primal–dual gap for PDHG, aPDHG and FISTA. It can be observed that the dynamic choice of the step sizes greatly improves the performance of the algorithm. It can also be observed that the fixed choice of step sizes for the PDHG algorithm seems to be fairly optimal for a certain accuracy, but for higher accuracy the performance of the algorithm breaks down. We can also see that in terms of the primal–dual gap – which in turn bounds the ℓ_2-error to the true solution – the aPDHG algorithm seems to be superior to the FISTA algorithm.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Example 5.7 (TV-deblurring). Here we show how to use the PDHG algorithm to minimize the image deblurring problem presented in Example 2.2. The TV-deblurring problem is given by

$$\min_u \lambda\|\mathrm{D}u\|_{2,1} + \frac{1}{2}\|a * u - u^\diamond\|^2,$$

where $u^\diamond$ is the $m \times n$ image from Example 2.2, a is a given blur kernel and $*$ denotes the two-dimensional convolution operation (using symmetric boundary conditions). For notational simplicity we will ignore the fact that the image $u^\diamond$ can be a colour image, as in Figure 2.2, and present the details about the algorithm only in the grey-scale case. The generalization to colour images is straightforward and is left to the reader. To simplify the notation further, we replace the convolution operator a with an equivalent linear operator A such that $Au = a * u$.

We will now describe two possible ways to minimize the TV-deblurring objective function using the PDHG algorithm. In the first method, called 'PD-explicit', we apply the convex conjugate to the total variation term (as in the ROF model), yielding the saddle-point problem

$$\min_u \max_{\mathbf{p}} \langle \mathrm{D}u, \mathbf{p}\rangle + \frac{1}{2}\|Au - u^\diamond\|^2 - \delta_{\{\|\cdot\|_{2,\infty}\le\lambda\}}(\mathbf{p}).$$

The PDHG algorithm could in principle be applied to this formulation if there were an efficient algorithm to compute the proximal map for the function $h(u) = \frac{1}{2}\|Au - u^\diamond\|^2$. A natural idea would be to compute this proximal map using the FFT (as in Example 5.9 below). However, here we want to use only convolution operations in the image domain. Hence, we restrict ourselves to the variant (5.6) of the PDHG algorithm, which can also deal with explicit gradients. The gradient with respect to the data term is easily computed as

$$\nabla g(u) = A^*(Au - u^\diamond),$$

where the adjoint operator A^* can be realized by a convolution with the adjoint kernel $\bar{a}$, which is the convolution kernel a rotated by 180 degrees. The Lipschitz constant of the gradient of $h(u)$ is computed as $L_h \le \|A\|^2 \le 1$. The proximal map with respect to the function $f^*(\mathbf{p}) = \delta_{\{\|\cdot\|_{2,\infty}\le\lambda\}}(\mathbf{p})$ is again computed using the projection formula (4.23). With this information, the primal–dual algorithm is easily implemented. According to (5.9), we set the step sizes as $\tau = 1/(L/c + L_h)$, and $\sigma = 1/(cL)$, for some $c > 0$ which yields a feasible pair of primal and dual step sizes.

In the second method, called 'PD-split', we apply the convex conjugate not only to the total variation term but also to the data term. This yields the saddle-point problem

$$\min_u \max_{\mathbf{p},q} \langle \mathrm{D}u, \mathbf{p}\rangle - \delta_{\{\|\cdot\|_{2,\infty}\le\lambda\}}(\mathbf{p}) + \langle Au, q\rangle - \frac{1}{2}\|q + u^\diamond\|^2,$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

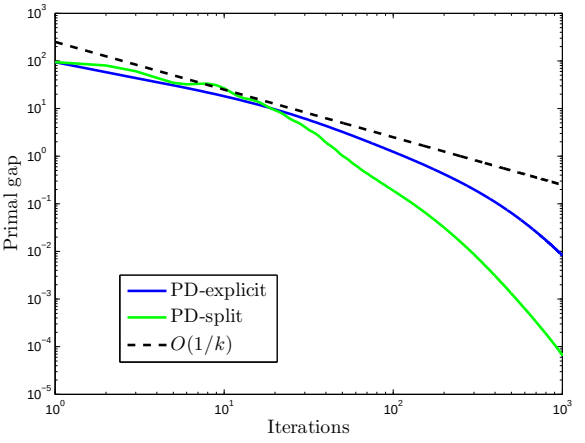


Figure 5.2. Minimizing the TV-deblurring problem applied to the image in Figure 2.2. We compare the performance of a primal–dual algorithm with explicit gradient steps (PD-explicit) and a primal–dual algorithm that uses a full splitting of the objective function (PD-split). PD-explicit seems to perform slightly better at the beginning, but PD-split performs better for higher accuracy.

where, letting $\mathbf{y} = (\mathbf{p}, q)$, $K^* = (\mathrm{D}^*, A^*)$, $f^*(\mathbf{y}) = (f^*_{\mathbf{p}}(\mathbf{p}), f^*_q(q))$, with

$$f^*_{\mathbf{p}}(\mathbf{p}) = \delta_{\{\|\cdot\|_{2,\infty} \le \lambda\}}(\mathbf{p}), \quad f^*_q(q) = \frac{1}{2}\|q + u^\diamond\|^2,$$

we obtain the saddle-point problem

$$\min_u \max_{\mathbf{y}} \langle Ku, \mathbf{y}\rangle - f^*(\mathbf{y}),$$

which exactly fits the class of problems that can be optimized by the PDHG algorithm. To implement the algorithm, we just need to know how to compute the proximal maps with respect to f^*. Since f^* is separable in $\mathbf{p}, q$, we can compute the proximal maps independently for both variables. The formula to compute the proximal map for $f^*_{\mathbf{p}}$ is again given by the projection formula (4.23). The proximal map for f^*_q requires us to solve pixelwise quadratic optimization problems. For a given $\tilde{q}$, its solution is given by

$$\hat{q} = \mathrm{prox}_{\sigma f^*_q}(\tilde{q}) \Leftrightarrow \hat{q}_{i,j} = \frac{\tilde{q}_{i,j} - \sigma u^\diamond_{i,j}}{1 + \sigma}.$$

We found it beneficial to apply a simple form of 2-block diagonal preconditioning by observing that the linear operator K is compiled from the two distinct but regular blocks D and A. According to Lemma 5.5, we can perform the following feasible choice of the step sizes: $\tau = c/(L + \sqrt{L_h})$, $\sigma_{\mathbf{p}} = 1/(cL)$, and $\sigma_q = 1/(c\sqrt{L_h})$, for some $c > 0$ where $\sigma_{\mathbf{p}}$ is used to update the $\mathbf{p}$ variable and σ_q is used to update the q variable.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Note that we cannot rely on the accelerated form of the PDHG algorithm because the objective function lacks strong convexity in u. However, the objective function is strongly convex in the variable Au, which can be used to achieve partial acceleration in the q variable (Valkonen and Pock 2015).

Figure 5.2 shows a comparison between the two different variants of the PDHG algorithm for minimizing the TV-deblurring problem from Example 2.2. In both variants we used $c = 10$. The true primal objective function has been computed by running the 'PD-split' algorithm for a large number of iterations. One can see that the 'PD-split' variant is significantly faster for higher accuracy. The reason is that the choice of the primal step size in 'PD-explicit' is more restrictive ($\tau < L_h$). On the other hand, 'PD-explicit' seems to perform well at the beginning and also has a smaller memory footprint.

Example 5.8 (minimizing the TV-ℓ_1 model). In the next example we will show how to minimize the TV-ℓ_1 model used in Example 2.3 for salt-and-pepper denoising. Let us recall the TV-ℓ_1 model for image denoising. It is given by the following non-smooth objective function:

$$\min_u \lambda \|\mathrm{D}u\|_{2,1} + \|u - u^\diamond\|_1,$$

where $u^\diamond$ is a given, noisy image of size $m \times n$ pixels. Using duality as in (3.10), we obtain the saddle-point problem

$$\min_u \max_{\mathbf{p}} \langle \mathrm{D}u, \mathbf{p}\rangle + \|u - u^\diamond\|_1 - \delta_{\{\|\cdot\|_{2,\infty} \le \lambda\}}(\mathbf{p}).$$

In order to implement the PDHG algorithm (or related proximal algorithms), we need to compute the proximal maps with respect to both the indicator function of the dual ball $f^*(\mathbf{p}) = \delta_{\{\|\cdot\|_{2,\infty} \le \lambda\}}(\mathbf{p})$ and the ℓ_1-norm in the data-fitting term $g(u) = \|u - u^\diamond\|_1$. The proximal map with respect to $f^*(\mathbf{p})$ is given (as in earlier examples) by the orthogonal projection onto independent 2-balls with radius λ (see (4.23)).

The proximal map with respect to g is given by the classical soft shrinkage operator. For a given $\tilde{u}$ and step size $\tau > 0$, the proximal map $\hat{u} = \mathrm{prox}_{\tau g}(\tilde{u})$ is given by

$$\hat{u}_{i,j} = u^\diamond_{i,j} + \max\{0, |\tilde{u}_{i,j} - u^\diamond_{i,j}| - \tau\} \cdot \mathrm{sgn}(\tilde{u}_{i,j} - u^\diamond_{i,j}).$$

Having detailed the computation of the proximal maps for the TV-ℓ_1 model, the implementation of the PDHG algorithm (Algorithm 6) is straightforward. The step size parameters were set to $\tau = \sigma = \|\mathrm{D}\|^{-1}$. For comparison, we also implemented the FBF algorithm (Tseng 2000) applied to the primal–dual system (5.5), which for the TV-ℓ_1 model and fixed step size is

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

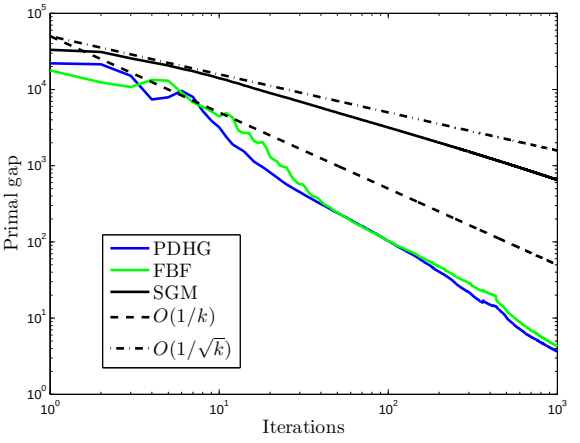


Figure 5.3. Minimizing the TV-ℓ_1 model applied to the image in Figure 2.3. The plot shows a comparison of the convergence of the primal gap between the primal–dual (PDHG) algorithm and the forward–backward–forward (FBF) algorithm. PDHG and FBF perform almost equally well, but FBF requires twice as many evaluations of the linear operator. We also show the performance of a plain subgradient method (SGM) in order to demonstrate the clear advantage of PDHG and FBF exploiting the structure of the problem.

given by

$$\begin{aligned} u^{k+1/2} &= \operatorname{prox}_{\tau g}(u^k - \tau \mathrm{D}^* \mathbf{p}^k), \\ \mathbf{p}^{k+1/2} &= \operatorname{prox}_{\tau f^*}(\mathbf{p}^k + \tau \mathrm{D} u^k), \\ u^{k+1} &= u^{k+1/2} - \tau \mathrm{D}^*(\mathbf{p}^{k+1/2} - \mathbf{p}^k), \\ \mathbf{p}^{k+1} &= \mathbf{p}^{k+1/2} + \tau \mathrm{D}(u^{k+1/2} - u^k). \end{aligned}$$

Observe that the FBF method requires twice as many matrix–vector multiplications as the PDHG algorithm. For simplicity, we used a fixed step size $\tau = \|\mathrm{D}\|^{-1}$. We also tested the FBF method with an Armijo-type line-search procedure, but it did not improve the results in this example.

Moreover, as a baseline, we also implemented a plain subgradient method (SGM), as presented in (4.10). In order to compute a subgradient of the total variation we used a Huber-type smoothing, but we set the smoothing parameter to a very small value, $\varepsilon = 10^{-30}$. For the subgradient of the data term, we just took the sign of the argument of the ℓ_1-norm. We used a diminishing step size of the form $c/\sqrt{k}$ for some $c > 0$ since it gave the best results in our experiments.

Figure 5.3 shows the convergence of the primal gap, where we computed the 'true' value of the primal objective function by running the PDHG algorithm for a large number of iterations. It can be observed that both

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

proximal algorithms (PDHG and FBF) converge significantly faster than plain SGM. This shows that in non-smooth optimization it is essential to exploit the structure of the problem. Observe that PDHG and FBF are faster than their theoretical worst-case rate $O(1/k)$ and PDHG seems to be slightly faster than FBF. Moreover, PDHG requires us to compute only half of the matrix–vector products of FBF.

5.2. Extensions

Convergence of more general algorithms of the same form are found in many papers in the literature: the subgradient can be replaced with general monotone operators (Vũ 2013*a*, Boţ *et al.* 2015, Davis and Yin 2015). In particular, some acceleration techniques carry on to this setting, as observed in Boţ *et al.* (2015). Davis and Yin (2015) discuss a slightly different method with similar convergence properties and rates which mix the cases of subgradients and monotone operators.

As for the case of forward–backward descent methods, this primal–dual method (being a variant of a proximal-point method) can be over-relaxed in some cases, or implemented with inertial terms, yielding better convergence rates (Chambolle and Pock 2015*a*).

Another important extension involves the Banach/non-linear setting. The proximity operators in (5.6) can be computed with non-linear metrics such as in the mirror prox algorithm (4.40). It dates back at least to Nemirovski (2004) in an extragradient form. For the form (5.6), it can be found in Hohage and Homann (2014) and is also implemented in Yanez and Bach (2014) to solve a matrix factorization problem. For a detailed convergence analysis see Chambolle and Pock (2015*a*).

Finally we should mention important developments towards optimal rates: Valkonen and Pock (2015) show how to exploit partial strong convexity (with respect to some of the variables) to gain acceleration, and obtain a rate which is optimal in both smooth and non-smooth situations; see also Chen *et al.* (2014*a*).

A few extensions to non-convex problems have recently been proposed (Valkonen 2014, Möllenhoff, Strekalovskiy, Moeller and Cremers 2015); see Section 6 for details.

5.3. Augmented Lagrangian type techniques

Perhaps one of the 'oldest' and most discussed approaches to solving non-smooth convex problems of the form (3.9), and beyond, is the 'alternating directions method of multipliers', or 'ADMM', proposed by Glowinski and Marroco (1975), studied by Gabay and Mercier (1976), and revisited many times; see for instance Boyd *et al.* (2011) for a review. It is also closely related to the 'split-Bregman' algorithm (Goldstein and Osher 2009, Zhang,

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 10 ADMM.

Choose $\gamma > 0$, y^0, z^0.
for all $k \geq 0$ **do**
  Find x^{k+1} by minimizing $x \mapsto f(x) - \langle z^k, Ax\rangle + \frac{\gamma}{2}\|b - Ax - By^k\|^2$,
  Find y^{k+1} by minimizing $y \mapsto g(y) - \langle z^k, By\rangle + \frac{\gamma}{2}\|b - Ax^{k+1} - By\|^2$,
  Update $z^{k+1} = z^k + \gamma(b - Ax^{k+1} - By^{k+1})$.
end for

Burger, Bresson and Osher 2010), which is inspired by Bregman iterations (Bregman 1967) and whose implementation boils down to an instance of the ADMM (though with interesting interpretations). A fairly general description of the relationships between the ADMM and similar splitting methods can be found in Esser (2009).

In its standard form, the ADMM aims at tackling constrained problems of the form

$$\min_{Ax+By=b} f(x) + g(y), \tag{5.15}$$

which become (3.9) if $b = 0$, $A = \mathrm{Id}$ and $B = -K$. The idea is to introduce a Lagrange multiplier z for the constraint, and write the global problem as a saddle-point optimization for an 'augmented Lagrangian' (Hestenes 1969, Powell 1969, Fortin and Glowinski 1982):

$$\min_{x,y} \sup_{z} f(x) + g(y) + \langle z, b - Ax - By\rangle + \frac{\gamma}{2}\|b - Ax - By\|^2,$$

where $\gamma > 0$ is a parameter. While it is obvious that the addition of the last quadratic term will not modify the optimality conditions or the value of the saddle-point, it greatly helps to stabilize the iterations by usually making the minimization problems in x, y (for fixed z) coercive, and hopefully solvable.

In practice, the most straightforward way to tackle this saddle-point problem should be to perform Uzawa's iterations, or an Arrow–Hurwicz-type method as in Section 5.1. Uzawa's algorithm would require us to minimize a problem in (x, y), for fixed z, which might be as difficult as the initial problem. When this is possible, the iteration $z^{k+1} = z^k + \gamma(b - Ax^{k+1} - By^{k+1})$ is then precisely an ascent method for the dual energy, whose gradient is $(1/\gamma)$-Lipschitz. But in general we need a different strategy. The idea which was proposed and analysed in Glowinski and Marroco (1975) and Gabay and Mercier (1976) simply consists in alternating the descent steps in x and y before updating z, as summarized in Algorithm 10. A related method called 'AMA' (Tseng 1991) drops the quadratic term in the first minimization. This can be interesting when f has strong convexity properties, but we will not discuss this variant. Many studies of the ADMM approach are to be found in the literature, from the thesis of Eckstein (1989) to more recent

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

convergence analyses (with rates) such as that of Nishihara *et al.* (2015) (Shefi and Teboulle 2014, He and Yuan 2015*b*, He and Yuan 2015*c*, Goldstein, O'Donoghue, Setzer and Baraniuk 2014). Some of these studies (Shefi and Teboulle 2014, Davis and Yin 2014*b*) relate the algorithm, and linearized variants, to the primal–dual approach discussed in Section 5.1; we will see indeed that these methods belong to the same class.

We now show briefly how this relationship is established, explaining the reason why convergence for the ADMM is ensured and the rates of convergence that one can expect from this algorithm without further tuning. Of course, this provides only a rough understanding of the method. A first remark is as follows. If we let

$$\tilde f(\xi) := \min_{\{x\,:\,Ax=\xi\}} f(x), \quad \tilde g(\eta) := \min_{\{y\,:\,By=\eta\}} g(y),$$

and set these to $+\infty$ when the set of constraints is empty, then these functions are convex, l.s.c., proper and the convex conjugates of $f^*(A^*\cdot)$ and $g^*(B^*\cdot)$, respectively; see Rockafellar (1997, Corollary 31.2.1).[13] Then one can rewrite the iterations of Algorithm 10, letting $\xi^k = Ax^k$ and $\eta^k = Ay^k$, in the form

$$\begin{aligned} \xi^{k+1} &= \operatorname{prox}_{\tilde f/\gamma}\left(b + \frac{z^k}{\gamma} - \eta^k\right),\\ \eta^{k+1} &= \operatorname{prox}_{\tilde g/\gamma}\left(b + \frac{z^k}{\gamma} - \xi^{k+1}\right),\\ z^{k+1} &= z^k + \gamma(b - \xi^{k+1} - \eta^{k+1}). \end{aligned} \tag{5.16}$$

In fact it is generally impossible to express the functions $\tilde f$ and $\tilde g$ explicitly, but the fact that the algorithm is computable implicitly assumes that the operators $\operatorname{prox}_{\tau\tilde f}, \operatorname{prox}_{\tau\tilde g}$ are computable. Observe that from the last two steps, thanks to Moreau's identity (3.8), we have

$$\frac{z^{k+1}}{\gamma} = b + \frac{z^k}{\gamma} - \xi^{k+1} - \operatorname{prox}_{\tilde g/\gamma}\left(b + \frac{z^k}{\gamma} - \xi^{k+1}\right) = \frac{1}{\gamma}\operatorname{prox}_{\gamma\tilde g^*}\big(z^k + \gamma(b - \xi^{k+1})\big).$$

Hence, letting $\tau = \gamma$, $\sigma = 1/\gamma$, $\bar z^k = z^k + \gamma(b - \xi^k - \eta^k)$, we see that the iterations (5.16) can be rewritten as

$$\begin{aligned} \xi^{k+1} &= \operatorname{prox}_{\sigma\tilde f}(\xi^k + \sigma\bar z^k),\\ z^{k+1} &= \operatorname{prox}_{\tau\tilde g^*}\big(z^k - \tau(\xi^{k+1} - b)\big),\\ \bar z^{k+1} &= 2z^{k+1} - z^k, \end{aligned} \tag{5.17}$$

[13] In infinite dimensions, we must require for instance that f^* is continuous at some point $A^*\zeta$; see in particular Bouchitté (2006).

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

which is exactly the primal–dual iterations (5.3)–(5.4) for the saddle-point problem

$$\min_{\xi} \max_{z} \tilde{f}(\xi) - \tilde{g}^*(z) + \langle z, \xi - b \rangle.$$

A consequence is that Theorem 5.1 applies, and allows us to derive a convergence rate of the form

$$\tilde{f}(\Xi^k) - \tilde{g}^*(z) + \langle z, \Xi^k - b \rangle - \left(\tilde{f}(\xi) - \tilde{g}^*(Z^k) + \langle Z^k, \xi - b \rangle\right) \leq \frac{1}{k}\left(\gamma \|\xi - \xi^0\|^2 + \frac{1}{\gamma}\|z - z^0\|\right),$$

where (Ξ^k, Z^k) are appropriate averages of the quantities (ξ^{i+1}, z^i), $i = 1, \ldots, k$. In practice, of course, we will have $\Xi^k = AX^k$, where X^k is the average of the points $(x^i)_{i=1}^k$. In addition, since each $\xi^i = Ax^i$ is obtained by a minimization of $f(x)$+(terms which depend only on Ax), we have $f(x^i) = \tilde{f}(\xi^i)$. Finally, the bound obtained in Theorem 5.1 is, in fact, at the step before being a bound on $\tilde{f}(\Xi^k)$, a bound on the quantity

$$\frac{1}{k}\sum_{i=1}^{k} \tilde{f}(\xi^i) = \frac{1}{k}\sum_{i=1}^{k} f(x^i) \geq f(X^k).$$

Hence the estimate above can be slightly improved and written in the following form, which involves the original functions f, g (we also use $\tilde{g}^*(z) = g^*(B^*z)$):

$$f(X^k) - g^*(B^*z) + \langle z, AX^k - b \rangle - \left(f(x) - g^*(B^*Z^k) + \langle Z^k, Ax - b \rangle\right) \leq \frac{1}{k}\left(\gamma \|Ax - Ax^0\|^2 + \frac{1}{\gamma}\|z - z^0\|\right).$$

Whether this can be turned into an effective bound for the energy of the initial problem depends on the particular functions f, g: one can obtain something useful only if an *a priori* bound on the z or ξ which reaches the supremum is known. For instance, if $\tilde{g}^*(z) = g^*(B^*z)$ has a globally bounded domain (equivalently, if $\tilde{g}$ has linear growth), one can take the sup over z in the estimate and get

$$f(X^k) + \tilde{g}(AX^k - b) - \left(f(x^*) + \tilde{g}(Ax^* - b)\right) \leq O\left(\frac{1}{k}\right),$$

where x^* is a minimizer for the problem. Similarly, if we can show a bound for the x which realizes the sup of the left-hand side, then we obtain a bound on the dual objective. Choosing a solution $z = z^*$, it follows that

$$f^*(A^*Z^k) + g^*(B^*Z^k) + \langle Z^k, b \rangle - \left(f^*(A^*z^*) + g^*(B^*z^*) + \langle z^*, b \rangle\right) \leq O\left(\frac{1}{k}\right).$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

These are ergodic rates, (slower) convergence rates in the non-ergodic sense are discussed in Davis and Yin (2014*a*); see also He and Yuan (2015*b*). This form of the ADMM has been generalized to problems involving more than two blocks (with some structural conditions) (He and Yuan 2015*a*, Fu, He, Wang and Yuan 2014) and/or to non-convex problems (see the references in Section 6.3).

Accelerated ADMM

The relationship that exists between the two previous method also allows us to derive accelerated variants of the ADMM method if either the function $\tilde{g}^*(z) = g^*(B^*z)$ or the function $\tilde{f}$ is strongly convex. The first case will occur when g has L_g-Lipschitz gradient and B^* is injective; then it will follow that $\tilde{g}^*$ is $1/(L_g\|(BB^*)^{-1}\|)$-strongly convex. This should not cover too many interesting cases, except perhaps the cases where $B = \mathrm{Id}$ and g is smooth so that the problem reduces to

$$\min_x f(x) + g(b - Ax),$$

which could then be tackled by a more standard accelerated descent method as in Section 4.7.[14] The second case corresponds to the case where f is itself μ_f-strongly convex:[15] here, $\tilde{f}^* = f^*(A^*\cdot)$ will have an $(\|A\|^2/\mu_f)$-Lipschitz gradient so that $\tilde{f}$ is itself $(\mu_f/\|A\|^2)$-strongly convex. This is the case for problem (2.6), for example.

When ∇g is L_g-Lipschitz and $B = \mathrm{Id}$ (so that $\tilde{g}^* = g^*$ is $(1/L_g)$-convex), the accelerated variant of (5.17), according to Algorithm 8, would be

$$\begin{aligned}
\xi^{k+1} &= \mathrm{prox}_{\sigma\tilde{f}}(\xi^k + \sigma_k \bar{z}^k),\\
z^{k+1} &= \mathrm{prox}_{\tau\tilde{g}^*}\big(z^k - \tau_k(\xi^{k+1} - b)\big),\\
\theta_k &= 1/\sqrt{1 + \frac{\tau_k}{L_g}}, \quad \tau_{k+1} = \theta_k\tau_k, \quad \sigma_{k+1} = 1/\tau_{k+1},\\
\bar{z}^{k+1} &= z^{k+1} + \theta_k(z^{k+1} - z^k).
\end{aligned}$$

[14] However, if we have a fast solver for the prox of $\tilde{g}$, it might still be interesting to consider the ADMM option.

[15] If both cases occur, then of course one must expect linear convergence, as in the previous section (Theorem 5.4). A derivation from the convergence of the primal–dual algorithm is found in Tan (2016), while general linear rates for the ADMM in smooth cases (including with over-relaxation and/or linearization) are proved by Deng and Yin (2016).

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

This, in turn, can be rewritten in the following 'ADMM'-like form, letting $\xi^k = Ax^k$, $\eta^k = y^k$, and $\tau_k = \gamma_k$:

$$\begin{aligned} x^{k+1} &= \arg\min_x f(x) - \langle z^k, Ax\rangle + \frac{\gamma_k}{2}\|b - Ax - y^k\|^2, \\ y^{k+1} &= \arg\min_y g(y) - \langle z^k, y\rangle + \frac{\gamma_k}{2}\|b - Ax^{k+1} - y\|^2, \\ z^{k+1} &= z^k + \gamma_k(b - Ax^{k+1} - y^{k+1}), \\ \gamma_{k+1} &= \gamma_k/\sqrt{1 + \frac{\gamma_k}{L_g}}. \end{aligned} \tag{5.18}$$

On the other hand, if f is μ_f-strongly convex, so that $\tilde{f}$ is ($\mu := \mu_f/\|A\|^2$)-strongly convex (indeed, $\tilde{f}^*(z) = f^*(A^*z)$ will have an ($\|A\|^2/\mu_f$)-Lipschitz gradient), we must revert to the formulation of the algorithm described in Appendix C.2. We start from the primal–dual formulation

$$\begin{aligned} \xi^{k+1} &= \mathrm{prox}_{\tau_k \tilde{f}}(\xi^k + \tau_k \bar{z}^k), \\ z^{k+1} &= \mathrm{prox}_{\sigma_k \tilde{g}^*}\big(z^k - \sigma_k(\xi^{k+1} - b)\big), \\ \bar{z}^{k+1} &= z^{k+1} + \theta_{k+1}(z^{k+1} - z^k), \end{aligned}$$

where $\tau_k, \sigma_k, \theta_k$ need to satisfy (C.4)–(C.5) (with μ_g replaced by μ) and (C.6), which reads as[16] $\theta_k^2 \sigma_k \tau_k \le 1$; see Appendix C.2. Then, this is in turn converted into an ADMM formulation, as before: letting

$$\eta^{k+1} = \arg\min_\eta \tilde{g}(\eta) + \frac{\sigma_k}{2}\|\eta + \xi^{k+1} - b\|^2 - \langle z^k, \eta\rangle,$$

we have, thanks again to Moreau's identity,

$$z^{k+1} = z^k - \sigma_k(\xi^{k+1} + \eta^{k+1} - b),$$

so that $\bar{z}^{k+1} = z^{k+1} - \theta_{k+1}\sigma_k(\xi^{k+1} + \eta^{k+1} - b)$. In particular, the argument in the prox defining ξ^{k+1} turns out to be $\xi^k + \tau z^k - \tau_k\theta_k\sigma_{k-1}(\xi^k + \eta^k - b)$ and we can obtain an ADMM formulation again provided that $\theta_k\tau_k\sigma_{k-1} = 1$ for all k, which in particular implies, using (C.5), that $\theta_k^2\tau_k\sigma_k = 1$, which shows (C.6). A possible choice is to consider equality in (C.4): together with (C.5) we deduce after a few calculations that, for each k, we should choose

$$\sigma_k = \frac{\mu + \sqrt{\mu^2 + 4\sigma_{k-1}^2}}{2}, \tag{5.19}$$

after having chosen an initial $\sigma_0 > \mu$. (We also have $\theta_k = \sqrt{1 - \mu/\sigma_k}$, $\tau_0 = 1/(\sigma_0 - \mu)$, $\tau_k = 1/(\theta_k^2\sigma_k) = 1/(\sigma_k - \mu)$, but these are not really needed

[16] As $L_h = 0$ and $L = 1$.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

in the final expressions.) We obtain the following ADMM algorithm:

$$x^{k+1} = \arg\min_x f(x) - \langle z^k, Ax\rangle + \frac{\sigma_k - \mu}{2}\|b - Ax - By^k\|^2, \tag{5.20}$$

$$y^{k+1} = \arg\min_y g(x) - \langle z^k, By\rangle + \frac{\sigma_k}{2}\|b - Ax^{k+1} - By\|^2, \tag{5.21}$$

$$z^{k+1} = z^k + \sigma_k(b - Ax^{k+1} - By^{k+1}). \tag{5.22}$$

Clearly from (5.19), $\sigma_k \geq \sigma_0 + k\mu/2$, so that (see (C.7)) we have

$$T_k \geq k + \frac{\mu}{4\sigma_0}k(k-1).$$

Interpreting (C.10) correctly, we obtain a control with a rate $O(1/k^2)$ on a gap which, depending on the particular problem, can be turned into a control on the energy. One issue with this kind of algorithm is the choice of the initial parameter σ_0, which can have a lot of influence on the effective convergence (Nishihara *et al.* 2015). Knowledge of the order of magnitude of $\|x_0 - x^*\|, \|z_0 - z^*\|$ and precise knowledge of μ might help to improve this choice.

In Figure 5.4 we compare the convergence of the primal–dual gap of ADMM and accelerated ADMM (aADMM) for the ROF model (2.6) applied to the image in Example 2.1. It can be observed that the advantage of the accelerated ADMM takes effect especially for higher accuracy. For comparison, we also plot the convergence of the accelerated primal–dual algorithm (aPDHG). One can observe that the ADMM algorithms are fast, especially at the beginning. Note, however, that one iteration of ADMM is computationally much more expensive than one iteration of aPDHG.

A few accelerated ADMM implementations have been proposed in the recent literature. A first important contribution is by Goldstein *et al.* (2014), who discuss various cases, and in the case considered here propose a heuristic approach which might be more efficient in practice. Powerful extensions of the techniques described in this section (and the next), leading to optimal mixed rates (depending on the structure of each function), are detailed in the recent contribution by Ouyang *et al.* (2015), where it seems the kind of acceleration described in this section first appeared.

Linearized ADMM

An important remark of Chambolle and Pock (2011), successfully exploited by Shefi and Teboulle (2014) to derive new convergence rates, is that the 'PDHG' primal–dual algorithm (5.3)–(5.4), is exactly the same as a linearized variant of the ADMM for $B = \mathrm{Id}$, with the first minimization step replaced by a proximal descent step (following a general approach introduced in Chen and Teboulle 1994),

$$x^{k+1} = \arg\min_x f(x) - \langle z^k, Ax\rangle + \frac{\gamma}{2}\|b - Ax - y^k\|^2 + \frac{\gamma}{2}\|x - x^k\|_M^2, \tag{5.23}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

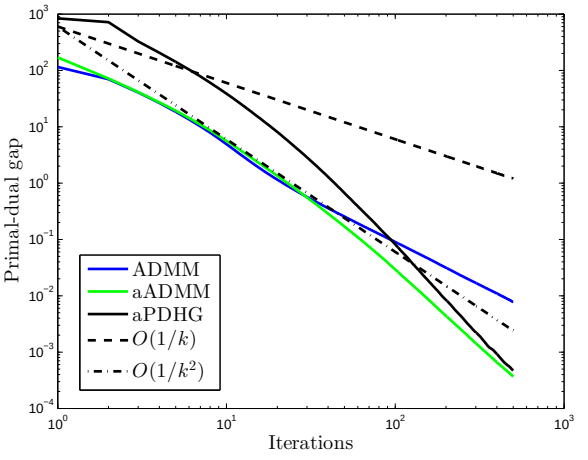


Figure 5.4. Comparison of ADMM and accelerated ADMM (aADMM) for solving the ROF model applied to the image in Figure 2.1. For comparison we also plot the convergence of the accelerated primal–dual algorithm (aPDHG). The ADMM methods are fast, especially at the beginning.

for M the preconditioning matrix

$$M = \frac{1}{\lambda}\,\mathrm{Id} - A^*A$$

which is positive semidefinite when $\lambda\|A\|^2 \leq 1$. Indeed, we obtain

$$x^{k+1} = \mathrm{prox}_{\frac{\lambda}{\gamma}f}\left(x^k + \frac{\lambda}{\gamma}A^*z^k - \lambda A^*(Ax^k + y^k - b)\right). \tag{5.24}$$

However, since $z^k = z^{k-1} + \gamma(b - Ax^k - y^k)$, letting $\sigma = \lambda/\gamma$, this equation becomes

$$x^{k+1} = \mathrm{prox}_{\sigma f}(x^k + \sigma A^*(2z^k - z^{k-1})),$$

while the second equation from (5.17) reads, using $\tilde{g} = g$,

$$z^{k+1} = \mathrm{prox}_{\tau g^*}(z^k - \tau(Ax^{k+1} - b)).$$

We recover precisely the PDHG algorithm of Theorem 5.1. Observe in particular that the condition $\sigma\tau\|A\|^2 = \lambda\|A\|^2 < 1$ is precisely the condition that makes M positive definite.

The natural extension of this remark consists in observing that if one can solve problem (5.23) for a preconditioning matrix M with $M + A^*A \geq 0$, then everything works the same, the only difference being that the proximity operator in (5.24) is now computed in the metric $M + A^*A$. This is particularly useful if g itself is quadratic (that is, $g(x) = \|Bx - x_0\|^2/2$), as it implies that one can replace the exact minimization of the problem in x

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

with a few iterations of a linear solver, and in many cases the output will be equivalent to exactly (5.23) in some (not necessarily known) metric M with $M + A^*A + (1/\gamma)K^*K \geq 0$. (For example, this occurs in the 'split Bregman' algorithm (Goldstein and Osher 2009), for which it has been observed, and proved by Zhang, Burger and Osher (2011), that one can do only one inner iteration of a linear solver; see also Yin and Osher (2013), who study inexact implementations.) For a precise statement we refer to Bredies and Sun (2015*b*, Section 2.3). It is shown there and in Bredies and Sun (2015*a*, 2015*c*) that careful choice of a linear preconditioner can lead to very fast convergence. A generalization of the ADMM in the same flavour is considered in Deng and Yin (2016), and several convergence rates are derived in smooth cases.

5.4. Douglas–Rachford splitting

Last but not least, we must mention another splitting method that is of the same class as the ADMM and PDHG algorithms. Observe that if we apply the PDHG algorithm to a problem with form (3.9), where $K = \mathrm{Id}$, then the iterations reduce to

$$x^{k+1} = \mathrm{prox}_{\tau g}(x^k - \tau y^k),$$
$$y^{k+1} = \mathrm{prox}_{\sigma f^*}(y^k + \sigma(2x^{k+1} - x^k)),$$

with the condition $\tau\sigma \leq 1$. We now assume $\sigma = 1/\tau$. Thanks to Moreau's identity (3.8),

$$2x^{k+1} - x^k + \tau y^k = \tau y^{k+1} + \mathrm{prox}_{\tau f}(2x^{k+1} - x^k + \tau y^k).$$

Letting $v^k = x^k - \tau y^k$ for each k, we find that the iterations can be equivalently written as

$$x^{k+1} = \mathrm{prox}_{\tau g} v^k, \tag{5.25}$$
$$v^{k+1} = v^k - x^{k+1} + \mathrm{prox}_{\tau f}(2x^{k+1} - v^k). \tag{5.26}$$

This is precisely the 'Douglas–Rachford' (DR) splitting algorithm (Douglas and Rachford 1956), in the form given in Lions and Mercier (1979, Algorithm II). Convergence is established in that paper, in the case where ∂f and ∂g are replaced with general maximal-monotone operators A and B. It can be seen as a particular way of writing the PDHG algorithm of Section 5.1, as for the ADMM. In fact, it is well known from Gabay (1983) and Glowinski and Le Tallec (1989) that the ADMM is the same as the Douglas–Rachford splitting implemented on the dual formulation of the problem; see also Setzer (2011) and Esser (2009) for other connections to similar algorithms. This method has been studied by many authors. An important observation is the fact that it can be written as a proximal-point algorithm: Eckstein and Bertsekas (1992) have shown and used this equivalence in particular to

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

generalize the algorithm to inexact minimizations. This follows quite easily from the analysis in this paper, since it is derived from the PDHG, which is an instance of the proximal-point algorithm when written in the form (5.5).

The fact that the convergence rate is improved when some of the functions or operators are smooth was mentioned by Lions and Mercier (1979); see Davis and Yin (2014*b*) for a detailed study which includes all the splitting described in this work. However, better rates can be achieved by appropriate accelerated schemes: in fact the same remarks which apply to the ADMM obviously apply here. Another obvious consequence is the fact that the acceleration techniques derived for the primal–dual algorithm can also be transferred to this splitting at the expense of some easy calculations – and this also holds for general monotone operators (Boţ *et al.* 2015). It might be simpler in practice, though, to write the iterations in the ADMM or primal–dual form before implementing acceleration tricks.

In a different direction, Briceño-Arias and Combettes (2011) have suggested applying this splitting to the saddle-point formulation (3.15) of problem (3.9), solving the inclusion $0 \in Az + Bz$ ($z = (x, y)^T$) with $A : (x, y) \mapsto (\partial g(x), \partial f^*(y))^T$ and $B : (x, y) \mapsto (K^*y, -Kx)^T$. This seems to produce excellent results at a reasonable cost; see in particular the applications to deconvolution in O'Connor and Vandenberghe (2014). A fairly general analysis of this approach is found in in Bredies and Sun (2015*b*), with applications to problems (2.6) and (7.4) in Bredies and Sun (2015*c*).[17]

Example 5.9 (TV-deblurring via ADMM or DR splitting). Let us turn back to the image deblurring problem from Example 2.2. For notational simplicity, we will again describe the problem for a grey-scale image u. One possible way to implement the TV-regularized deblurring problem (2.7) is to write the minimization problem as follows (letting $Au := a * u$):

$$\min_u \lambda\|\mathrm{D}u\|_{2,1} + \frac{1}{2}\|Au - u^\diamond\|^2 = \min_{\mathbf{p}} \lambda\|\mathbf{p}\|_{2,1} + G(\mathbf{p}),$$

where $\mathbf{p} = (p_1, p_2)$ and

$$G(\mathbf{p}) := \min_{u:\mathrm{D}u=\mathbf{p}} \frac{1}{2}\|Au - u^\diamond\|^2$$

(and $+\infty$ if $\mathbf{p}$ is not a discrete gradient). Observe that the prox of G can be computed as $\hat{\mathbf{p}} = \mathrm{prox}_{\tau G}(\tilde{\mathbf{p}})$ if and only if $\hat{\mathbf{p}} = \mathrm{D}u$, where u solves

$$\min_u \frac{1}{2\tau}\|\mathrm{D}u - \tilde{\mathbf{p}}\|^2 + \frac{1}{2}\|Au - u^\diamond\|^2.$$

[17] We will discuss acceleration strategies in the spirit of Theorem 5.2 in a forthcoming paper.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) TV (PSNR ≈ 26.6) (b) Huber (PSNR ≈ 28.0)

Figure 5.5. Solving the image deblurring problem from Example 2.2. (a) Problem (2.7) after 150 iterations of Douglas–Rachford (DR) splitting. (b) Huber variant after 150 iterations with accelerated DR splitting. The figure shows that after the same number of iterations, the accelerated algorithm yields a higher PSNR value.

It follows that $\hat{\mathbf{p}}$ must be given by

$$\hat{\mathbf{p}} = \mathrm{D}(\mathrm{D}^*\mathrm{D} + \tau A^*A)^{-1}(\mathrm{D}^*\tilde{\mathbf{p}} + \tau A^*u^\diamond).$$

When A corresponds to a convolution operator, then this computation can be efficiently implemented using an FFT. Since this approach implicitly assumes that the image is periodic, we have pre-processed the original image by bringing its intensity values to the average values at the image boundaries (see Example 2.2). Another natural and clever approach to deal with this boundary issue, suggested by Almeida and Figueiredo (2013), is to replace the data term with $\frac{1}{2}\|M(a * u - u^\diamond)\|^2$, where M is a diagonal mask which is zero in a strip, of width the size of the support of a, along the boundaries, and 1 in the middle rectangle (or in other words, we now use $Au := Ma * u$ where u is defined on a larger grid than $u^\diamond$, and can now be assumed to be periodic). This modification improves the results, but observe that it then requires about twice as many FFTs per iteration.

The proximity operator is defined by

$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p}} \lambda\|\mathbf{p}\|_{2,1} + \frac{1}{2\tau}\|\mathbf{p} - \tilde{\mathbf{p}}\|^2,$$

and since $\mathbf{p} = (\mathbf{p}_{1,1}, \dots, \mathbf{p}_{m,n})$, its solution is given by a pixelwise shrinkage operation,

$$\hat{\mathbf{p}}_{i,j} = \left(1 - \frac{1}{\max\{1, \frac{1}{\tau\lambda}|\tilde{\mathbf{p}}_{i,j}|_2\}}\right)\tilde{\mathbf{p}}_{i,j}.$$

As the prox of both $\lambda\|\mathbf{p}\|_{2,1}$ and $G(\mathbf{p})$ are solvable, one can implement a DR splitting for this problem. Of course, the reader expert in these approaches

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

will see right away that it leads to exactly the same computations as those we would perform for an ADMM implementation (Esser 2009, Zhang *et al.* 2011, Zhang *et al.* 2010, Getreuer 2012). In practice, we alternate (5.25) and (5.26) with g replaced with $\|\cdot\|_{2,1}$ and f with G defined above.

As already observed, the total variation regularizer often produces unwanted staircasing (flat regions) in the reconstructed image, as it promotes sparse gradients. Hence, for such a problem, the 'Huber' regularizer (4.18) will usually produce equivalent if not better results. The idea is to replace the function $\|\mathbf{p}\|_{2,1}$ with its Huber variant $H_\varepsilon(\mathbf{p})$, for $\varepsilon > 0$, where h_ε is given in (4.20). An important advantage is that the new function now has a $1/\varepsilon$-Lipschitz gradient, so acceleration is possible (Algorithm 8). We obtain essentially the same results as before with far fewer iterations (or better results with the same number of iterations): see Figure 5.5 (the PSNR values are computed only in a central area, where the image is not smoothed before deblurring).

6. Non-convex optimization

In this very incomplete section, we mention some extensions of the methods described so far to non-convex problems. Of course, many interesting optimization problems in imaging are not convex. If f is a smooth non-convex function, many of the optimization methods designed for smooth convex functions will work and find a critical point of the function. For instance, a simple gradient method (4.1) always guarantees that, denoting $g^k = \nabla f(x^k)$,

$$\begin{aligned} f(x^{k+1}) &= f(x^k - \tau g^k) \\ &= f(x^k) - \tau\langle \nabla f(x^k), g^k\rangle + \int_0^\tau (\tau - t)\langle D^2 f(x^k - tg^k)g^k, g^k\rangle \,\mathrm{d}t \\ &\leq f(x^k) - \tau\left(1 - \frac{\tau L}{2}\right)\|g^k\|^2 \end{aligned}$$

as long as $D^2 f \leq L\,\mathrm{Id}$, whether positive or not. Hence, if $0 < \tau < 2/L$, then $f(x^k)$ will still be decreasing. If f is coercive and bounded from below, we deduce that subsequences of $(x^k)_k$ converge to some critical point. Likewise, inertial methods can be used and are generally convergent (Zavriev and Kostyuk 1991) if ∇f is L-Lipschitz and with suitable assumptions which ensure the boundedness of the trajectories.

6.1. Non-convex proximal descent methods

Now, what about non-smooth problems? A common way to extend accelerated methods of this kind to more general problems is to consider problems

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 11 'iPiano' algorithm (Ochs *et al.* 2014).

Choose $x^0, x^{-1} \in \mathcal{X}$, and for all k, parameters $\beta \in [0,1)$, $\alpha \in (0, 2(1-\beta)/L)$.

for all $k \geq 0$ **do**

$$x^{k+1} = \mathrm{prox}_{\alpha g}(x^k - \alpha \nabla f(x^k) + \beta(x^k - x^{k-1})) \tag{6.1}$$

end for

of the form (4.26) with f still being smooth but not necessarily convex. Then one will generally look for a critical point (hoping of course that it might be optimal!) by trying to find x^* such that

$$\nabla f(x^*) + \partial g(x^*) \ni 0.$$

There is a vast literature on optimization techniques for such problems, which have been tackled in this form at least since Mine and Fukushima (1981) and Fukushima and Mine (1981). These authors study and prove the convergence of a proximal FB descent (combined with an approximate line-search in the direction of the new point) for non-convex f. Recent contributions in this direction, in particular for imaging problems, include those of Grasmair (2010), Chouzenoux, Pesquet and Repetti (2014), Bredies, Lorenz and Reiterer (2015*a*) and Nesterov (2013). We will describe the inertial version of Ochs, Chen, Brox and Pock (2014), which is of the same type but seems empirically faster, which is natural to expect as it reduces to the standard heavy ball method (Section 4.8.2) in the smooth case. Let us describe the simplest version, with constant steps: see Algorithm 11. Here again, L is the Lipschitz constant of ∇f. Further, subsequences of $(x^k)_k$ will still converge to critical points of the energy; see Ochs *et al.* (2014, Theorem 4.8). This paper also contains many interesting variants (with varying steps, monotonous algorithms, *etc.*), as well as convergence rates for the residual of the method.

6.2. Block coordinate descent

We must mention another particular form of problem (4.26) that is interesting for applications in the non-convex case, for instance for matrix factorization problems, or problems where the product of two variables appears in a (smooth) term of the objective. This takes the form

$$\min_{x,y} f(x,y) + g_1(x) + g_2(y), \tag{6.2}$$

where f is again smooth but not necessarily convex, while g_1, g_2 are non-smooth and simple functions, possibly non-convex.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Algorithm 12 'PALM' algorithm (Bolte *et al.* 2014).

Choose $(x^0, y^0) \in \mathcal{X} \times \mathcal{Y}$, $\gamma_1 > 1$, $\gamma_2 > 1$.
for all $k \geq 0$ **do**
Choose $\tau_{1,k} = (\gamma_1 L_1(y^k))^{-1}$ and let

$$x^{k+1} = \operatorname{prox}_{\tau_{1,k} g_1}(x^k - \tau_{1,k} \nabla_x f(x^k, y^k)). \tag{6.3}$$

Choose $\tau_{2,k} = (\gamma_2 L_2(x^{k+1}))^{-1}$ and let

$$y^{k+1} = \operatorname{prox}_{\tau_{2,k} g_2}(y^k - \tau_{2,k} \nabla_y f(x^{k+1}, y^k)). \tag{6.4}$$

end for

The convergence of alternating minimizations or proximal (implicit) descent steps in this setting (which is not necessarily covered by the general approach of Tseng 2001) has been studied by Attouch *et al.* (2013), Attouch, Bolte, Redont and Soubeyran (2010) and Beck and Tetruashvili (2013). However, Bolte, Sabach and Teboulle (2014) have observed that, in general, these alternating steps will not be computable. These authors propose instead to alternate linearized proximal descent steps, as shown in Algorithm 12. Here, $L_1(y)$ is the Lipschitz constant of $\nabla_x f(\cdot, y)$, while $L_2(x)$ is the Lipschitz constant of $\nabla_y f(x, \cdot)$. These are assumed to be bounded from below[18] and above (in the original paper the assumptions are slightly weaker). Also, for convergence one must require that a minimizer exists; in particular, the function must be coercive.

Then it is proved by Bolte *et al.* (2014, Lemma 5) that the distance of the iterates to the set of critical points of (6.2) goes to zero. Additional convergence results are shown if, in addition, the objective function has a very generic 'KL' property. We have presented a simplified version of the PALM algorithm: in fact, there can be more than two blocks, and the simple functions g_i need not even be convex: as long as they are bounded from below, l.s.c., and their proximity operator (which is possibly multivalued, but still well defined by (3.6)) can be computed, then the algorithm will converge. We use an inertial variant of PALM (Pock and Sabach 2016) in Section 7.12 to learn a dictionary of patches.

6.3. Saddle-point-type methods

In the same way, primal–dual first-order methods, including the ADMM, can easily be extended to non-convex optimization and one could mention an infinite number of papers where this has been suggested; see for instance the

[18] A bound from below does not make any sense for the Lipschitz constants, but here it corresponds to a bound from above for the steps $\tau_{i,k}$.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

references in Hong, Luo and Razaviyayn (2015). Some structural conditions which guarantee convergence to a critical point (of the Lagrangian) are given in a few recent papers (Chartrand and Wohlberg 2013, Magnússon, Chathuranga Weeraddana, Rabbat and Fischione 2014, Hong *et al.* 2015, Wang, Yin and Zeng 2015), sometimes involving more than two blocks. For instance, with such an algorithm one can easily tackle the non-convex variant of (2.6),

$$\min_u \lambda\varphi(\mathrm{D}u) + \frac{1}{2}\|u - u^\diamond\|^2,$$

with $\varphi(\cdot) = \|\cdot\|_{p,q}$ for $q \in [0, 1)$ and either $p = 2$ or $p = q$, or another similar non-convex function with sublinear growth. This formulation dates back at least to Geman and Geman (1984), Geman and Reynolds (1992) and Blake and Zisserman (1987) (which motivated Mumford and Shah 1989). However, it is often considered that the benefit of using such a formulation is not obvious with respect to the computationally simpler model (2.6). The deblurring model (2.7) can be extended in the same way.

It is clear that the ADMM or PDHG algorithms are still possible to implement for this problem, once we know how to compute the proximity operator

$$\bar{\mathbf{p}} \mapsto \arg\min_{\mathbf{p}} \varphi(\mathbf{p}) + \frac{1}{2\tau}\|\mathbf{p} - \bar{\mathbf{p}}\|^2,$$

at least for sufficiently large τ. Here, an interesting remark is that if the Hessian of φ is bounded from below (φ is 'semiconvex'), then this problem is strongly convex for sufficiently small τ and has a unique solution. This is the case considered in Magnússon *et al.* (2014) or Möllenhoff *et al.* (2015), for example. However, the method should also converge if this is not satisfied in a bounded set, which is the case for q-norms with $q < 1$ (Wang *et al.* 2015): in this case we should select one solution of the proximity operator, which is now possibly multivalued (yet still a monotone operator). In general this operator is not explicitly computable (except for $p = 0$ – hard thresholding – or $p = 1/2$), but its solution can be approximated by simple methods (*e.g.* Newton).

Another interesting non-convex extension of saddle-point methods is Valkonen's generalization of the PDHG method (Valkonen 2014), in which the linear operator K is replaced with a non-linear operator; this is particularly useful for applications to diffusion tensor imaging in MRI.

Example 6.1 (solving non-convex TV-deblurring). Here we extend Example 5.9 by replacing the total variation regularizer with a non-convex variant. We consider the problem

$$\min_u \lambda\varphi(\mathrm{D}u) + \frac{1}{2}\|Au - u^\diamond\|^2. \tag{6.5}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

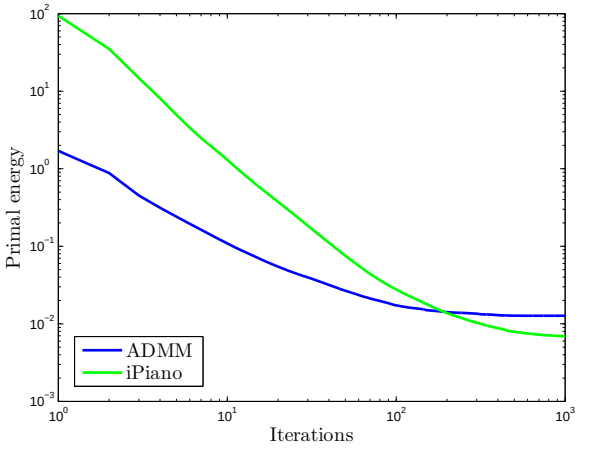


Figure 6.1. Image deblurring using a non-convex variant of the total variation. The plot shows the convergence of the primal energy for the non-convex TV model using ADMM and iPiano. In order to improve the presentation in the plot, we have subtracted a strict lower bound from the primal energy. ADMM is faster at the beginning but iPiano finds a slightly lower energy.

The non-convex regularizer $\varphi(\mathbf{p})$ with $\mathbf{p} = (\mathbf{p}_{1,1}, \dots, \mathbf{p}_{m,n})$ is given by

$$\varphi(\mathbf{p}) = \frac{1}{2} \sum_{i,j} \ln\left(1 + \frac{|\mathbf{p}_{i,j}|_2^2}{\mu^2}\right),$$

where $\mu > 0$ is a parameter. In what follows, we consider two different approaches to minimizing the non-convex image deblurring problem. In our first approach we consider a non-convex extension of the ADMM algorithm, and hence we restrict ourselves to exactly the same setting as in Example 5.9. Observe that the proximity operator of $\tau\lambda\varphi$ is the (unique, if τ is sufficiently small) solution of

$$\frac{\tau\lambda}{\mu^2} \frac{\mathbf{p}_{i,j}}{1 + \frac{|\mathbf{p}_{i,j}|_2^2}{\mu^2}} + \mathbf{p}_{i,j} = \bar{\mathbf{p}}_{i,j}$$

(for all pixels i, j), which we can compute here using a fixed point (Newton) iteration, or by solving a third-order polynomial.

The second approach is based on directly minimizing the primal objective using the iPiano algorithm (Algorithm 11). We perform a forward–backward splitting by taking explicit steps with respect to the (differentiable) regularizer $f(u) = \lambda\varphi(\mathrm{D}u)$, and perform a backward step with respect to the data term $g(u) = \frac{1}{2}\|Au - u^\diamond\|^2$. The gradient with respect to the regularization

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) NC, TV, ADMM (PSNR ≈ 27.80)

(b) NC, TV, iPiano (PSNR ≈ 27.95)

(c) NC, learned, iPiano (PSNR ≈ 29.56)

(d) 7×7 filters

Figure 6.2. Image deblurring using non-convex functions after 150 iterations. (a, b) Results of the non-convex TV-deblurring energy obtained from ADMM and iPiano. (c) Result obtained from the non-convex learned energy, and (d) convolution filters D_k sorted by their corresponding λ_k value (in descending order) used in the non-convex learned model. Observe that the learned non-convex model leads to a significantly better PSNR value.

term is given by

$$\nabla f(u) = \frac{\lambda}{\mu^2} \mathrm{D}^* \tilde{\mathbf{p}},$$

where $\tilde{\mathbf{p}}$ is of the form $\tilde{\mathbf{p}} = (\tilde{\mathbf{p}}_{1,1}, \dots, \tilde{\mathbf{p}}_{m,n})$, and

$$\tilde{\mathbf{p}}_{i,j} = \frac{(\mathrm{D}u)_{i,j}}{1 + \frac{|(\mathrm{D}u)_{i,j}|_2^2}{\mu^2}}.$$

The gradient is Lipschitz-continuous with Lipschitz constant

$$L \leq \frac{\lambda}{\mu^2} \|\mathrm{D}\|^2 \leq \frac{8\lambda}{\mu^2}.$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

The proximal map with respect to the data term $g(u)$ can be easily implemented using the FFT. We used the following parameter settings for the iPiano algorithm: $\beta = 0.7$ and $\alpha = 2(1-\beta)/L$.

Moreover, we implemented a variant of (6.5), where we have replaced the non-convex TV regularizer with a learned regularizer of the form

$$\sum_{k=1}^{K} \lambda_k \varphi(D_k u),$$

where the parameters $\lambda_k > 0$, and the linear operators D_k (convolution filters, in fact) are learned from natural images using bilevel optimization. We used the 48 filters of size 7×7 obtained by Chen *et al.* (2014*b*); see also Figure 6.2. We again minimize the resulting non-convex objective function using the iPiano algorithm.

Figure 6.1 compares the performance of the ADMM algorithm with that of the iPiano algorithm, using the image from Example 2.2. We used the parameter settings $\lambda = 1/5000$ and $\mu = 0.1$. One can observe that ADMM converges faster but iPiano is able to find a slightly lower energy. This is explained by the ability of the iPiano algorithm to overcome spurious stationary points by making use of an inertial force. Figure 6.2 shows the results obtained from the non-convex learned regularizer. The PSNR values show that the learned regularizer leads to significantly better results, of course at a higher computational cost.

7. Applications

In the rest of the paper we will show how the algorithms presented so far can be used to solve a number of interesting problems in image processing, computer vision and learning. We start by providing some theoretical background on the total variation and some extensions.

7.1. Total variation

In the continuous setting, the idea of the ROF model is to minimize the following energy in the space of functions with bounded variation:

$$\min_u \lambda \int_\Omega |Du| + \frac{1}{2} \int_\Omega (u(x) - u^\diamond(x))^2 \, \mathrm{d}x, \tag{7.1}$$

where Ω is the image domain, $u^\diamond$ is a given (noisy) image and $\lambda > 0$ is a regularization parameter. The term $\int_\Omega |Du|$ in the energy is the total variation (TV) of the image u and the gradient operator D is understood in its distributional sense. A standard way to define the total variation is

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

by duality, as follows, assuming $\Omega \subset \mathbb{R}^d$ is a d-dimensional open set:

$$\int_\Omega |Du| \tag{7.2}$$
$$:= \sup\left\{ -\int_\Omega u(x) \operatorname{div} \varphi(x)\, \mathrm{d}x : \varphi \in C_c^\infty(\Omega; \mathbb{R}^d),\ |\varphi(x)| \le 1,\ \forall x \in \Omega \right\}$$

and we say that u has bounded variation if and only if this quantity is finite. The space

$$\mathrm{BV}(\Omega) = \left\{ u \in L^1(\Omega) : \int_\Omega |Du| < +\infty \right\},$$

of functions with bounded variation, equipped with the norm $\|u\|_{\mathrm{BV}} = \|u\|_{L^1} + \int_\Omega |Du|$, is a Banach space; see Giusti (1984), Ambrosio, Fusco and Pallara (2000) or Evans and Gariepy (1992) for details. The function $|\cdot|$ could in fact be any norm, in which case the constraint on $\varphi(x)$ in (7.2) should use the polar norm

$$|\xi|^\circ := \sup_{|x| \le 1} \langle \xi, x \rangle$$

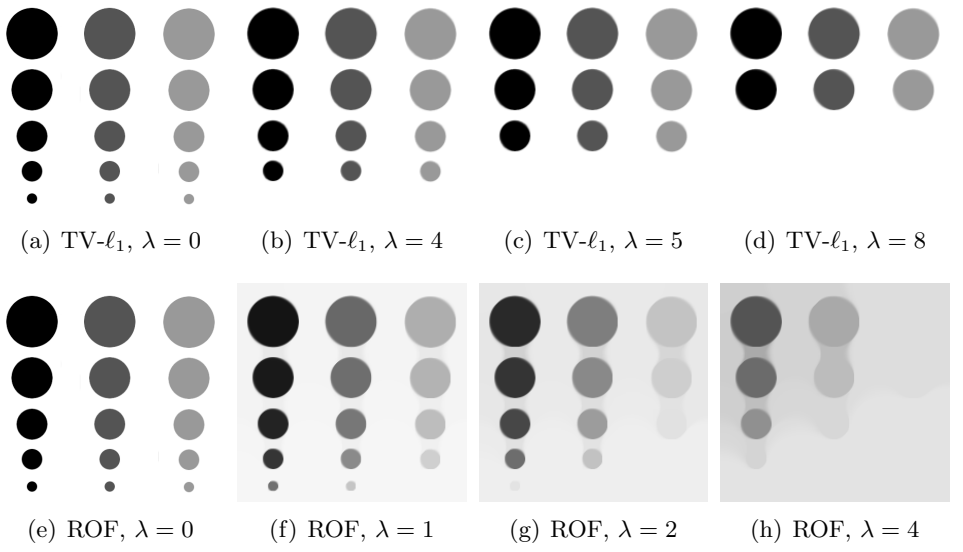
and read $|\varphi(x)|^\circ \le 1$ for all x. The most common choices (at least for grey-scale images) are (possibly weighted) 2- and 1-norms. The main advantage of the total variation is that it allows for sharp jumps across hypersurfaces, for example edges or boundaries in the image, while being a convex functional, in contrast to other Sobolev norms. For smooth images u we easily check from (7.2) (integrating by parts) that it reduces to the L^1-norm of the image gradient, but it is also well defined for non-smooth functions. For characteristic functions of sets it measures the length or surface of the boundary of the set inside Ω (this again is easy to derive, at least for smooth sets, from (7.2) and Green's formula). This also makes the total variation interesting for geometric problems such as image segmentation.

Concerning the data-fitting term, numerous variations of (7.1) have been proposed in the literature. A simple modification of the ROF model is to replace the squared data term with an L^1-norm (Nikolova 2004, Chan and Esedoḡlu 2005):

$$\min_u \lambda \int_\Omega |Du| + \int_\Omega |u(x) - u^\diamond(x)|\, \mathrm{d}x. \tag{7.3}$$

The resulting model, called the 'TV-ℓ_1 model', turns out to have interesting new properties. It is purely geometric in the sense that the energy decomposes on the level set of the image. Hence, it can be used to remove structures of an image of a certain scale, and the regularization parameter λ can be used for scale selection. The TV-ℓ_1 model is also effective in removing impulsive (outlier) noise from images.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) TV-ℓ_1, $\lambda = 0$ (b) TV-ℓ_1, $\lambda = 4$ (c) TV-ℓ_1, $\lambda = 5$ (d) TV-ℓ_1, $\lambda = 8$

(e) ROF, $\lambda = 0$ (f) ROF, $\lambda = 1$ (g) ROF, $\lambda = 2$ (h) ROF, $\lambda = 4$

Figure 7.1. Contrast invariance of the TV-ℓ_1 model. (a–d) Result of the TV-ℓ_1 model for varying values of the regularization parameter λ. (e–h) Result of the ROF model for varying values of λ. Observe the morphological property of the TV-ℓ_1 model. Structures are removed only with respect to their size, but independent of their contrast.

In the presence of Poisson noise, a popular data-fitting term (justified by a Bayesian approach) is given by the generalized Kullback–Leibler divergence, which is the Bregman distance of the Boltzmann–Shannon entropy (Steidl and Teuber 2010, Dupé, Fadili and Starck 2012). This yields the following TV-entropy model:

$$\min_{u(x)>0} \lambda \int_\Omega |Du| + \int_\Omega u(x) - u^\diamond(x) \log u(x)\, \mathrm{d}x.$$

This model has applications in synthetic aperture radar (SAR) imaging, for example.

We have already detailed the discretization of TV models in (2.6) and we have shown that an efficient algorithm to minimize total variation models is the PDHG algorithm (Algorithm 6 and its variants). A saddle point formulation of discrete total variation models that summarizes the different aforementioned data-fitting terms is as follows:

$$\min_u \max_{\mathbf{p}} \langle \mathrm{D}u, \mathbf{p}\rangle + g(u) - \delta_{\{\|\cdot\|_{2,\infty} \le \lambda\}}(\mathbf{p}),$$

where $\mathrm{D} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n\times 2}$ is the finite difference approximation of the gradient operator defined in (2.4), and $\mathbf{p} = (p_1, p_2) \in \mathbb{R}^{m\times n\times 2}$ is the dual variable. Let $u^\diamond \in \mathbb{R}^{m\times n}$ be a given noisy image, then for $g(u) = \frac{1}{2}\|u - u^\diamond\|^2$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

we obtain the ROF model, for $g(u) = \|u - u^\diamond\|_1$ we obtain the TV-ℓ_1 model and for $g(u) = \sum_{i,j} u_{i,j} - u^\diamond_{i,j} \log u_{i,j} + \delta_{(0,\infty)}(u)$ we obtain the TV-entropy model. The implementation of the models using the PDHG algorithm only differs in the implementation of the proximal operators $\hat{u} = \operatorname{prox}_{\tau g}(\tilde{u})$. For all $1 \le i \le m$, $1 \le j \le n$ the respective proximal operators are given by

$$\hat{u}_{i,j} = \frac{\tilde{u}_{i,j} + \tau u^\diamond_{i,j}}{1+\tau} \qquad \text{(ROF)},$$

$$\hat{u}_{i,j} = u^\diamond_{i,j} + \max\{0, |\tilde{u}_{i,j} - u^\diamond_{i,j}| - \tau\} \cdot \operatorname{sgn}(\tilde{u}_{i,j} - u^\diamond_{i,j}) \qquad \text{(TV-}\ell_1\text{)},$$

$$\hat{u}_{i,j} = \max\left\{0, \frac{\tilde{u}_{i,j} - \tau + \sqrt{(\tilde{u}_{i,j} - \tau)^2 + 4\tau u^\diamond_{i,j}}}{2}\right\} \qquad \text{(TV-entropy)}.$$

Figure 7.1 demonstrates the contrast invariance of the TV-ℓ_1 model and compares it to the ROF model. Both models were minimized using Algorithm 6 (PDHG) or Algorithm 8. Gradually increasing the regularization parameter λ in the TV-ℓ_1 model has the effect that increasingly larger structures are removed from the image. Observe that the structures are removed only with respect to their size and not with respect to their contrast. In the ROF model, however, scale and contrast are mixed such that gradually increasing the regularization parameter results in removing structures with increased size and contrast.

Figure 7.2 compares the ROF model with the TV-entropy model for image denoising in the presence of Poisson noise. The noisy image of size 480×640 pixels has been generated by degrading an aerial image of Graz, Austria, with Poisson noise with parameter the image values scaled between 0 and 50. Both models have been minimized using the PDHG algorithm. It can be observed that the TV-entropy model adapts better to the noise properties of the Poisson noise and hence leads to better preservation of dark structures and exhibits better contrast.

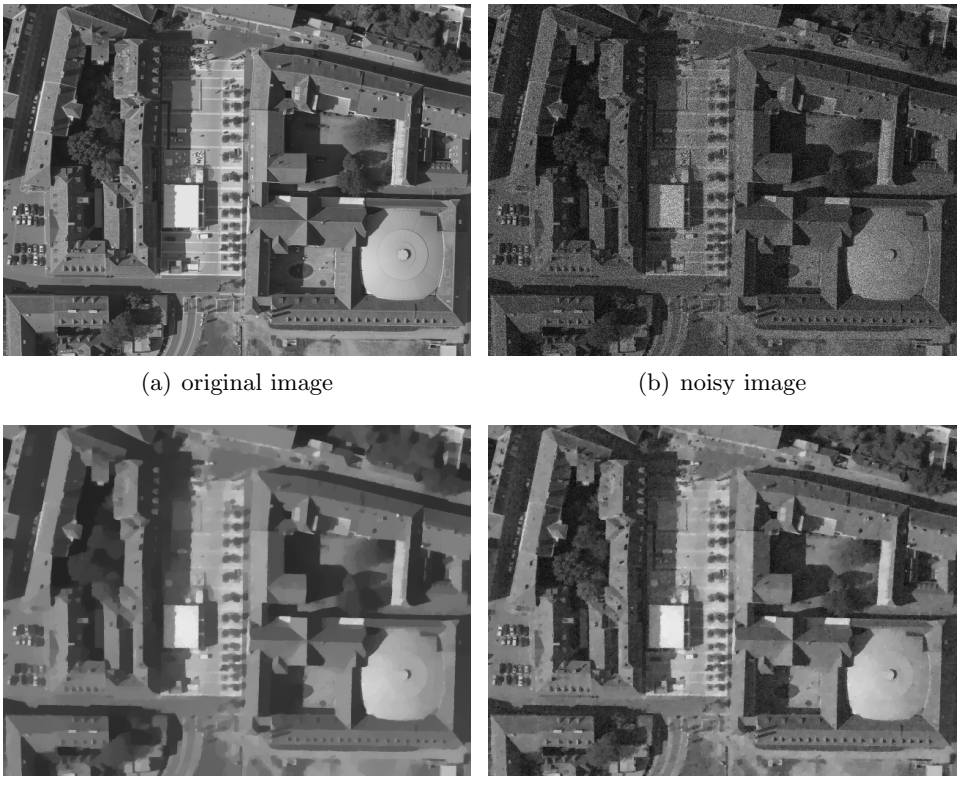
7.2. Total generalized variation

There have been attempts to generalize the total variation to higher-order smoothness, for example using the notion of infimal convolution (Chambolle and Lions 1995, 1997). One generalization is the so-called total generalized variation (TGV) proposed in Bredies, Kunisch and Pock (2010). An image denoising model based on second-order total generalized variation (TGV^2) is given by

$$\min_u \inf_v \lambda_1 \int_\Omega |Du - v| + \lambda_0 \int_\Omega |Dv| + \frac{1}{2} \int_\Omega (u(x) - u^\diamond(x))^2 \, \mathrm{d}x, \qquad (7.4)$$

where $u \in \mathrm{BV}(\Omega)$, $v \in \mathrm{BV}(\Omega; \mathbb{R}^2)$, and $\lambda_{0,1} > 0$ are tuning parameters. The idea of TGV^2 is to force the gradient Du of the image to deviate only on a

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) original image (b) noisy image

(c) ROF, $\lambda = 1/10$ (d) TV-entropy, $\lambda = 1/7$

Figure 7.2. Total variation based image denoising in the presence of Poisson noise. (a) Aerial view of Graz, Austria, (b) noisy image degraded by Poisson noise. (c) Result using the ROF model, and (d) result using the TV-entropy model. One can see that the TV-entropy model leads to improved results, especially in dark regions, and exhibits better contrast.

sparse set from a vector field v which itself has sparse gradient. This will get rid of the staircasing effect on affine parts of the image, while still preserving the possibility of having sharp edges. The discrete counterpart of (7.4) can be obtained by applying the same standard discretization techniques as in the case of the ROF model.

We introduce the discrete scalar images $u, u^\diamond \in \mathbb{R}^{m\times n}$ and vectorial image $\mathbf{v} = (v_1, v_2) \in \mathbb{R}^{m\times n\times 2}$. The discrete version of the TGV^2 model is hence given by

$$\min_{u,\mathbf{v}} \lambda_1 \|\mathrm{D}u - \mathbf{v}\|_{2,1} + \lambda_0 \|\mathbf{D}\mathbf{v}\|_{2,1} + \frac{1}{2}\|u - u^\diamond\|^2,$$

where $\mathbf{D} : \mathbb{R}^{m\times n\times 2} \to \mathbb{R}^{m\times n\times 4}$ is again a finite difference operator that computes the Jacobian (matrix) of the vectorial image $\mathbf{v}$, which we treat

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

as a vector here. It can be decomposed into $\mathbf{Dv} = (\mathrm{D}v_1, \mathrm{D}v_2)$, where D is again the standard finite difference operator introduced in (2.4). The discrete versions of the total first- and second-order variations are given by

$$\|\mathrm{D}u - \mathbf{v}\|_{2,1} = \sum_{i=1,j=1}^{m,n} \sqrt{((\mathrm{D}u)_{i,j,1} - v_{i,j,1})^2 + ((\mathrm{D}u)_{i,j,2} - v_{i,j,2})^2},$$

$$\|\mathbf{Dv}\|_{2,1} = \sum_{i=1,j=1}^{m,n} \sqrt{(\mathrm{D}v_1)_{i,j,1}^2 + (\mathrm{D}v_1)_{i,j,2}^2 + (\mathrm{D}v_2)_{i,j,1}^2 + (\mathrm{D}v_2)_{i,j,2}^2}.$$

In order to minimize the discrete TGV^2 model, we rewrite it as the saddle-point problem

$$\min_{u,\mathbf{v}} \max_{\mathbf{p},\mathbf{q}} \langle \mathrm{D}u - \mathbf{v}, \mathbf{p}\rangle + \langle \mathbf{Dv}, \mathbf{q}\rangle + \frac{1}{2}\|u - u^\diamond\|^2 - \delta_{\{\|\cdot\|_{2,\infty} \le \lambda_1\}}(\mathbf{p}) - \delta_{\{\|\cdot\|_{2,\infty} \le \lambda_0\}}(\mathbf{q}),$$

where $\mathbf{p} = (p_1, p_2) \in \mathbb{R}^{m\times n\times 2}$ and $\mathbf{q} = (q_1, q_2, q_3, q_4) \in \mathbb{R}^{m\times n\times 4}$ are the dual variables. Combining $(u, \mathbf{v})$, and $(\mathbf{p}, \mathbf{q})$ into the primal and dual vectors, we can see that the above saddle-point problem is exactly of the form (3.10), with

$$K = \begin{pmatrix} \mathrm{D} & -I \\ 0 & \mathbf{D} \end{pmatrix}, \quad g(u) = \frac{1}{2}\|u - u^\diamond\|^2,$$

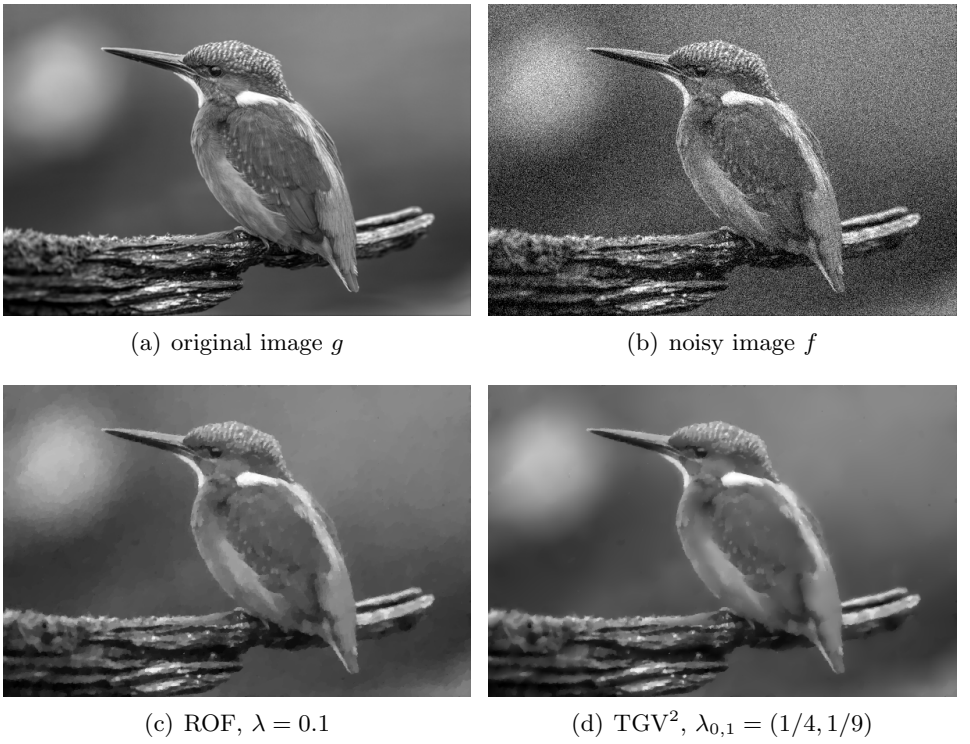
$$f^*(\mathbf{p}, \mathbf{q}) = \delta_{\{\|\cdot\|_{2,\infty} \le \lambda_1\}}(\mathbf{p}) + \delta_{\{\|\cdot\|_{2,\infty} \le \lambda_0\}}(\mathbf{q}).$$

The proximal map with respect to the data term $g(u)$ is the same as the proximal map of the ROF model. The proximal map with respect to $f^*(\mathbf{p}, \mathbf{q})$ reduces to projections onto the respective polar balls, as shown in (4.23). The Lipschitz constant L of the operator K can be estimated as $L = \|K\| \le \sqrt{12}$. With this information, the PDHG algorithm can be easily implemented. Figure 7.3 shows a qualitative comparison between TV and TGV^2 regularization using an image of size 399×600. One can see that TGV^2 regularization leads to better reconstruction of the smooth region in the background while still preserving the sharp discontinuities of the bird. TV regularization, on the other hand, suffers from the typical staircasing effect in the background region.

7.3. Vectorial total variation

Since the seminal paper 'Color TV' (Blomgren and Chan 1998), different extensions of the total variation to vector-valued images have been proposed. The most straightforward approach would be to apply the scalar total variation to each colour channel individually, but this clearly ignores the coupling between the colour channels. Assume we are given a vector-valued image $u = (u_1, \ldots, u_k)$ with k channels. A proper generalization of

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) original image g (b) noisy image f

(c) ROF, $\lambda = 0.1$ (d) TGV^2, $\lambda_{0,1} = (1/4, 1/9)$

Figure 7.3. Comparison of TV and TGV^2 denoising. (a) Original input image, and (b) noisy image, where we have added Gaussian noise with standard deviation $\sigma = 0.1$. (c) Result obtained from the ROF model, and (d) result obtained by minimizing the TGV^2 model. The main advantage of the TGV^2 model over the ROF model is that it is better at reconstructing smooth regions while still preserving sharp discontinuities.

the total variation to vector-valued images is now to define a suitable matrix norm that acts on the distributional Jacobian Du. Assume that our image domain Ω is a subset of $\mathbb{R}^d$, where d is the dimension of the image, and also assume for the moment that the image function u is sufficiently smooth that the Jacobian $J(x) = \nabla u(x)$ exists. An interesting class of matrix norms is given by p-Schatten norms, which are defined as

$$|J(x)|_{\mathcal{S}_p} = \left(\sum_{n=1}^{\min\{d,k\}} \sigma_n^p(J(x)) \right)^{1/p}, \quad \text{for all } p \in [1, \infty),$$
$$|J(x)|_{\mathcal{S}_\infty} = \max_{n \in \{1,\dots,\min\{d,k\}\}} \sigma_n(J(x)),$$

where the $\sigma_n(J(x))$ denote the singular values of the Jacobian $J(x)$ (*i.e.*, the square roots of the eigenvalues of $J(x)J(x)^*$ or $J(x)^*J(x)$).

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

If $p = 2$, the resulting norm is equivalent to the Frobenius norm, which corresponds to one of the most classical choices (Bresson and Chan 2008), though other choices might also be interesting (Sapiro and Ringach 1996, Chambolle 1994). For $p = 1$ (Duran, Möller, Sbert and Cremers 2016*b*, 2016*a*), the norm is equivalent to the nuclear norm and hence forces the Jacobian to be of low rank. The comparisons in Duran, Moeller, Sbert and Cremers (2016*a*) confirm the superior performance of this choice. If $p = \infty$, the resulting norm is the operator norm of the Jacobian, penalizing its largest singular value, which turns out to produce more spurious colours than the previous choice (Goldluecke, Strekalovskiy and Cremers 2012).

Using the dual formulation of the total variation (7.2), we can readily define a vectorial version of the total variation based on p-Schatten norms which is now valid also for images in the space of functions with bounded variation:

$$\int_\Omega |Du|_{\mathcal{S}_p} = \tag{7.5}$$
$$\sup\Big\{ -\int_\Omega u(x) \cdot \operatorname{div} \varphi(x)\,\mathrm{d}x : \varphi \in C^\infty(\Omega; \mathbb{R}^{d\times k}),\ |\varphi(x)|_{\mathcal{S}_q} \le 1,\ \forall x \in \Omega \Big\},$$

where q is the parameter of the polar norm associated with the parameter p of the Schatten norm and is given by $1/p + 1/q = 1$. Based on that we can define a vectorial ROF model as

$$\min_u \lambda \int_\Omega |Du|_{\mathcal{S}_p} + \frac{1}{2}\int_\Omega |u(x) - u^\diamond(x)|_2^2\,\mathrm{d}x. \tag{7.6}$$

The discretization of the vectorial ROF model is similar to the discretization of the standard ROF model. We consider a discrete colour image $\mathbf{u} = (u_r, u_g, u_b) \in \mathbb{R}^{m\times n\times 3}$, where $u_r, u_g, u_b \in \mathbb{R}^{m\times n}$ denote the red, green, and blue colour channels, respectively. We also consider a finite difference operator $\mathbf{D} : \mathbb{R}^{m\times n\times 3} \to \mathbb{R}^{m\times n\times 2\times 3}$ given by $\mathbf{Du} = (\mathrm{D}u_r, \mathrm{D}u_g, \mathrm{D}u_b)$, where D is again the finite difference operator defined in (2.4). The discrete colour ROF model based on the 1-Schatten norm is given by

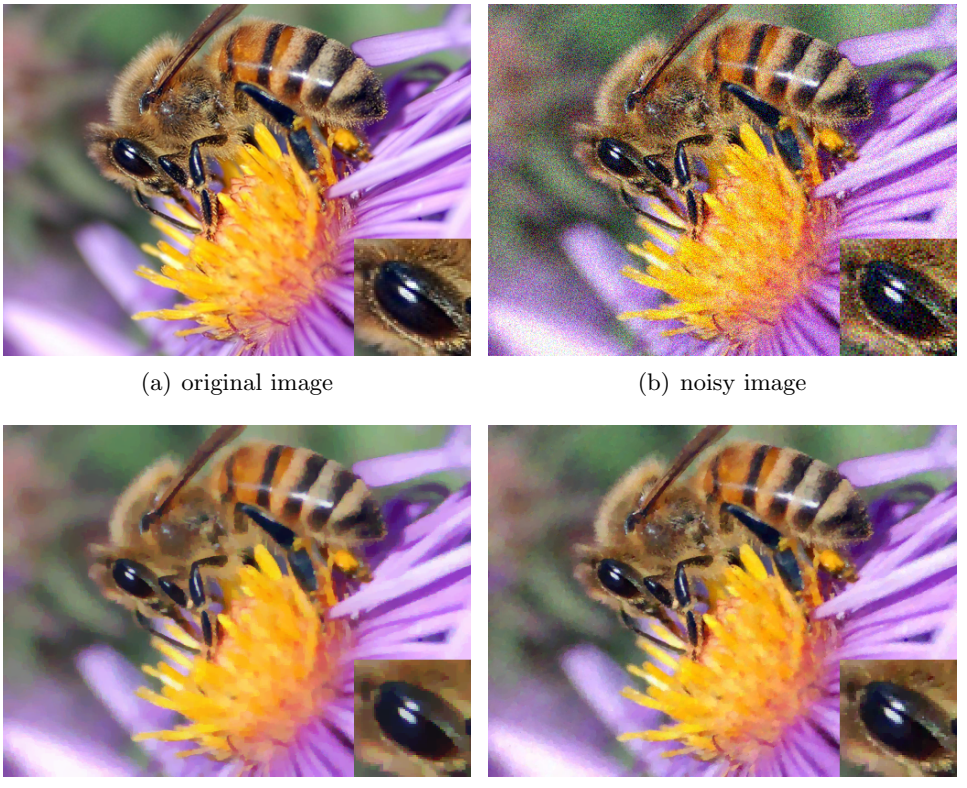
$$\min_{\mathbf{u}} \lambda\|\mathbf{Du}\|_{\mathcal{S}_1,1} + \frac{1}{2}\|\mathbf{u} - \mathbf{u}^\diamond\|^2.$$

The vectorial ROF model can be minimized either by applying Algorithm 5 to its dual formulation or by applying Algorithm 8 to its saddle-point formulation. Let us consider the saddle-point formulation:

$$\min_{\mathbf{u}} \max_{\mathbf{P}} \langle \mathbf{Du}, \mathbf{P}\rangle + \frac{1}{2}\|\mathbf{u} - \mathbf{u}^\diamond\|^2 + \delta_{\{\|\cdot\|_{\mathcal{S}_\infty,\infty} \le \lambda\}}(\mathbf{P}),$$

where $\mathbf{P} \in \mathbb{R}^{m\times n\times 2\times 3}$ is the tensor-valued dual variable, hence the dual variable can also be written as $\mathbf{P} = (\mathbf{P}_{1,1}, \ldots, \mathbf{P}_{m,n})$, where $\mathbf{P}_{i,j} \in \mathbb{R}^{2\times 3}$ is

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) original image (b) noisy image

(c) Frobenius, $\lambda = 0.1$ (d) nuclear, $\lambda = 0.1$

Figure 7.4. Denoising a colour image using the vectorial ROF model. (a) Original RGB colour image, and (b) its noisy variant, where Gaussian noise with standard deviation $\sigma = 0.1$ has been added. (c) Solution of the vectorial ROF model using the Frobenius norm, and (d) solution using the nuclear norm. In smooth regions the two variants lead to similar results, while in textured regions the nuclear norm leads to significantly better preservation of small details (see the close-up views).

a 2×3 matrix. Hence, the polar norm ball $\{\|\mathbf{P}\|_{\mathcal{S}_\infty,\infty} \leq \lambda\}$ is also given by

$$\{\mathbf{P} = (\mathbf{P}_{1,1}, \ldots, \mathbf{P}_{m,n}) : |\mathbf{P}_{i,j}|_{\mathcal{S}_\infty} \leq \lambda, \text{ for all } i, j\},$$

hence the set of variables $\mathbf{P}$, whose tensor-valued components $\mathbf{P}_{i,j}$ have an operator norm less than or equal to λ. To compute the projection to the polar norm ball we can use the singular value decomposition (SVD) of the matrices. Let U, S, V with $U \in \mathbb{R}^{2\times 2}$, let $S \in \mathbb{R}^{2\times 3}$ with $S = \operatorname{diag}(s_1, s_2)$, and let $V \in \mathbb{R}^{3\times 3}$ be an SVD of $\tilde{\mathbf{P}}_{i,j}$, that is, $\tilde{\mathbf{P}}_{i,j} = USV^T$. As shown by Cai, Candès and Shen (2010), the orthogonal projection of $\tilde{\mathbf{P}}_{i,j}$ to the polar norm ball $\{\|\mathbf{P}\|_{\mathcal{S}_\infty,\infty} \leq \lambda\}$ is

$$\Pi_{\{\|\cdot\|_{\mathcal{S}_\infty,\infty} \leq \lambda\}}(\tilde{\mathbf{P}}_{i,j}) = US_\lambda V^T, \quad S_\lambda = \operatorname{diag}(\min\{s_1, \lambda\}, \min\{s_2, \lambda\}).$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Figure 7.4 shows an example of denoising a colour image of size 384×512 with colour values in the range $[0, 1]^3$. It can be seen that the nuclear norm indeed leads to better results as it forces the colour gradients to be of low rank, promoting better correlation of the colour channels. This can be best observed in textured regions, where the nuclear norm is much better at preserving small details.

7.4. Total variation regularized linear inverse problems

The total variation and its generalizations (*e.g.* TGV) have also become a popular regularizer for general linear inverse problems such as image deconvolution and image reconstruction in computer tomography (CT) (Sidky, Kao and Pan 2006) or magnetic resonance imaging (MRI) (Knoll, Bredies, Pock and Stollberger 2011). The main idea for speeding up MRI reconstructions is via compressed sensing by sparsely sampling the data in the Fourier space. Using direct reconstruction from such undersampled data would clearly lead to strong artifacts. Therefore, a better idea is to consider a total variation regularized problem of the form

$$\min_u \lambda \int_\Omega |Du| + \sum_{c=1}^{C} \frac{1}{2} \|\mathcal{F}(\sigma_c u) - g_c\|_2^2, \tag{7.7}$$

where $\mathcal{F}$ denotes the Fourier transform, g_c, $c = 1, \dots, C$ are multiple channel data obtained from the coils, and σ_c are the corresponding complex-valued sensitivity estimates. Discretization of the model is straightforward. We consider a complex valued image $u \in \mathbb{C}^{m \times n}$ of size $m \times n$ pixels and also the usual finite difference approximation D of the gradient operator, as defined in (2.4). We also assume that we are given $c = 1, \dots, C$ discrete versions of the sensitivities $\sigma_c \in \mathbb{C}^{m \times n}$ and data $g_c \in \mathbb{C}^{m \times n}$. The discrete model is given by

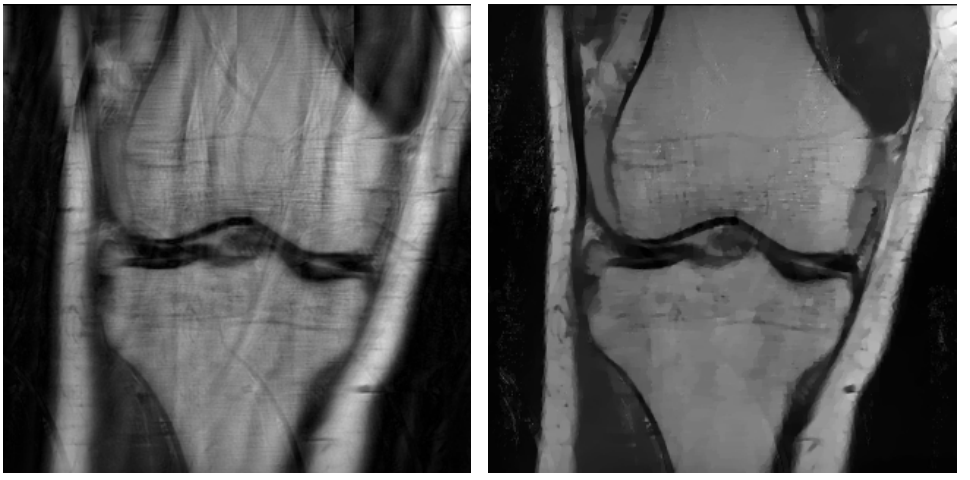
$$\min_u \lambda \|\mathrm{D}u\|_{2,1} + \sum_{c=1}^{C} \frac{1}{2} \|\mathrm{F}(\sigma_c \circ u) - g_c\|^2,$$

where $\mathrm{F} : \mathbb{C}^{m \times n} \to \mathbb{C}^{m \times n}$ denotes the (discrete) fast Fourier transform, and $\circ$ denotes the Hadamard product (the element-wise product of the two matrices). In order to minimize the TV-MRI objective function, we first transform the problem into a saddle-point problem:

$$\min_u \max_{\mathbf{p}} \langle \mathrm{D}u, \mathbf{p} \rangle + \sum_{c=1}^{C} \frac{1}{2} \|\mathrm{F}(\sigma_c \circ u) - g_c\|^2 - \delta_{\{\|\cdot\|_{2,\infty} \le \lambda\}}(\mathbf{p}),$$

where $\mathbf{p} \in \mathbb{C}^{m \times n \times 2}$ is the dual variable. Observe that we have just dualized

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) no regularization (b) TV regularization

Figure 7.5. TV regularized reconstruction of one slice of an MRI of a knee from partial Fourier data. (a) Least-squares reconstruction without using total variation regularization, and (b) reconstruction obtained from the total variation regularized reconstruction model.

the total variation term but kept the data-fitting term

$$h(u) = \sum_{c=1}^{C} \frac{1}{2} \|\mathrm{F}(\sigma_c \circ u) - g_c\|^2$$

explicitly. The gradient of this fitting term is given by

$$\nabla h(u) = \overline{\sigma_c} \circ \mathrm{F}^*(\mathrm{F}(\sigma_c \circ u) - g_c),$$

where F^* denotes the adjoint of the Fourier transform and $\overline{\sigma_c}$ denotes the complex conjugate of σ_c. Assuming that F is orthonormal and assuming that $\|\sigma_c\|_\infty \leq 1$ for all $c = 1, \ldots, C$, the Lipschitz constant of $\nabla h(u)$ is bounded by $L_h \leq 1$. Following Example 5.7 we can use the 'Condat–Vũ' variant (5.6) of the PDHG algorithm that can include explicit gradient steps.

Figure 7.5 shows the TV-MRI model for the reconstruction of a slice of an MRI image of a knee.[19] Figure 7.5(a) shows the reconstruction of the slice without using total variation regularization and Figure 7.5(b) shows the result based on total variation regularization. Observe that the total

[19] Data courtesy of Florian Knoll, Center for Biomedical Imaging and Center for Advanced Imaging Innovation and Research (CAI2R), Department of Radiology, NYU School of Medicine.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

variation regularized reconstruction model successfully removes the artifacts introduced by the missing Fourier data.

7.5. Optical flow

In computer vision, total variation regularization has become a very popular choice for optical flow computation. Optical flow is the apparent motion of intensity patterns (caused by objects, structures, surfaces) in a scene. The main underlying assumption of optical flow is the *brightness constancy assumption*, which states that the intensities of visual structures stay constant over time. Let $I(x,t)$ be a spatio-temporal intensity function (*e.g.* a video), defined on $\Omega \times [0,T]$, where Ω is a subset of $\mathbb{R}^d$ and $[0,T]$ with $T>0$ is the time domain. The brightness constancy assumption states that

$$I(x,t) = I(x+\Delta_x, t+\Delta_t),$$

for sufficiently small spatial displacements Δ_x and time differences Δ_t. Assuming the spatio-temporal intensity function is sufficiently smooth, a first-order Taylor expansion can be used to derive a linearized brightness constancy assumption, also known as the *optical flow constraint* (Lucas and Kanade 1981, Horn and Schunck 1981):

$$\nabla I(x,t)^T \cdot (v(x),1) = 0, \quad \text{for all } x,t \in \Omega \times [0,T],$$

where $v=(v_1,v_2)$ is the velocity field. Direct solution of this equation for v is heavily ill-posed. Indeed, the velocity can only be estimated in the direction of spatial image gradients (aperture problem), and homogeneous regions do not provide any information. If the brightness constancy assumption does not hold in practice, it can also be replaced with a gradient constancy or more sophisticated photo consistency metrics. It turns out that the total variation and its generalizations are effective regularizers for optical flow estimation since they force the flow field to be piecewise constant or piecewise smooth. A total variation based optical flow model is given by

$$\min_v \lambda \int_\Omega |Dv|_{p,1} + \frac{1}{q}\int_\Omega |\nabla I(x,t)^T \cdot (v(x),1)|^q \,\mathrm{d}x, \tag{7.8}$$

where different norms can be considered for both the total variation and the data-fitting term. The most common choice is $p=2$, and $q=1$ (Brox, Bruhn, Papenberg and Weickert 2004, Zach, Pock and Bischof 2007, Chambolle and Pock 2011). For numerical solution we discretize the TV-ℓ_1 optical flow model in the same spirit as we did with the previous TV models. We consider a discrete velocity field $\mathbf{v} = (v_1, v_2) \in \mathbb{R}^{m\times n\times 2}$, where v_1 corresponds to the horizontal velocity and v_2 corresponds to the vertical velocity. It can also be written in the form of $\mathbf{v} = (\mathbf{v}_{1,1}, \dots, \mathbf{v}_{m,n})$, where $\mathbf{v}_{i,j} = (v_{i,j,1}, v_{i,j,2})$ is the local velocity vector. To discretize the total

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

variation, we again consider a finite difference approximation of the vectorial gradient $\mathbf{D} : \mathbb{R}^{m\times n\times 2} \to \mathbb{R}^{m\times n\times 4}$, defined by $\mathbf{Dv} = (\mathrm{D}v_1, \mathrm{D}v_2)$, where D is defined in (2.4). In order to discretize the data term, we consider a certain point in time for which we have computed finite difference approximations for the space-time gradient of $I(x,t)$. It is necessary to have at least two images in time in order to compute the finite differences in time. We denote the finite difference approximation to the space-time gradient by $\mathbf{r} = (r_1, r_2, r_3) \in \mathbb{R}^{m\times n\times 3}$; it also has the structure $\mathbf{r} = (\mathbf{r}_{1,1}, \ldots, \mathbf{r}_{m,n})$, where $\mathbf{r}_{i,j} = (r_{i,j,1}, r_{i,j,2}, r_{i,j,3})$ is the space-time gradient at pixel i,j. The discrete model is then given by

$$\min_{\mathbf{v}} \lambda\|\mathbf{Dv}\|_{2,1} + \sum_{i=1,j=1}^{m,n} |\mathbf{r}_{i,j}\cdot(\mathbf{v}_{i,j},1)|,$$

where

$$\mathbf{r}_{i,j}\cdot(\mathbf{v}_{i,j},1) = r_{i,j,1}v_{i,j,1} + r_{i,j,2}v_{i,j,2} + r_{i,j,3}.$$

For the vectorial total variation we consider the standard 2-vector norm, that is,

$$\|\mathbf{Dv}\|_{2,1} = \sum_{i=1,j=1} \sqrt{(\mathrm{D}v_1)^2_{i,j,1} + (\mathrm{D}v_1)^2_{i,j,2} + (\mathrm{D}v_2)^2_{i,j,1} + (\mathrm{D}v_2)^2_{i,j,2}}.$$

The TV-ℓ_1 model is non-smooth, and hence we introduce dual variables $\mathbf{p} \in \mathbb{R}^{m\times n\times 4}$ and consider its saddle-point formulation

$$\min_{\mathbf{v}} \max_{\mathbf{p}} \langle \mathbf{Dv}, \mathbf{p}\rangle + \sum_{i=1,j=1}^{m,n} |\mathbf{r}_{i,j}\cdot(\mathbf{v}_{i,j},1)| - \delta_{\{\|\cdot\|_{2,\infty}\le\lambda\}}(\mathbf{p}),$$

which can be solved using Algorithm 6 (PDHG). It remains to detail the solution of the proximal map with respect to the function

$$g(\mathbf{v}) = \sum_{i=1,j=1}^{m,n} |\mathbf{r}_{i,j}\cdot(\mathbf{v}_{i,j},1)|.$$

A simple computation (Zach *et al.* 2007) shows that the proximal map is given by

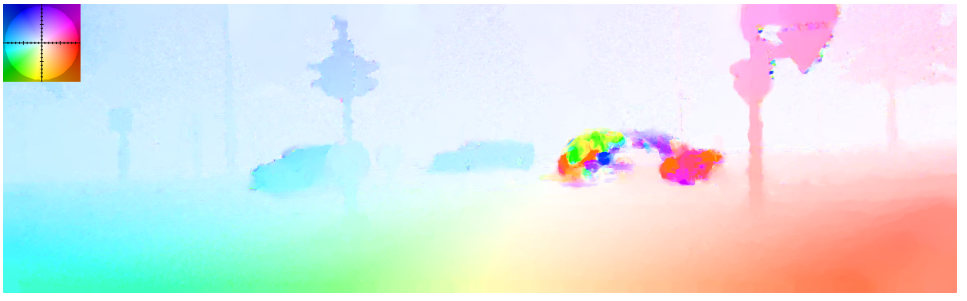
$$\hat{\mathbf{v}} = \mathrm{prox}_{\tau g}(\tilde{\mathbf{v}}) \Leftrightarrow$$

$$\hat{\mathbf{v}}_{i,j} = \tilde{\mathbf{v}}_{i,j} + \begin{cases} \tau\mathbf{r}_{i,j} & \text{if } \mathbf{r}_{i,j}\cdot(\tilde{\mathbf{v}}_{i,j},1) < -\tau|\mathbf{r}_{i,j}|^2, \\ -\tau\mathbf{r}_{i,j} & \text{if } \mathbf{r}_{i,j}\cdot(\tilde{\mathbf{v}}_{i,j},1) > \quad \tau|\mathbf{r}_{i,j}|^2, \\ -(\mathbf{r}_{i,j}\cdot(\tilde{\mathbf{v}}_{i,j},1)/|\mathbf{r}_{i,j}|^2)\,\mathbf{r}_{i,j} & \text{else.} \end{cases}$$

With this information, the PDHG algorithm can be easily implemented. Since the optical flow constraint is valid only for small spatial displacements, the minimization of the TV-ℓ_1 optical flow model is usually embedded within

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) input images (averaged)



(b) velocity field

Figure 7.6. Optical flow estimation using total variation. (a) A blending of the two input images. (b) A colour coding of the computed velocity field. The colour coding of the velocity field is shown in the upper left corner of the image.

a coarse-to-fine warping framework based on image pyramids (Brox *et al.* 2004). Figure 7.6 shows an example of computing the optical flow in a typical car driving scene.[20] It can be observed that the motion field has been recovered nicely, but we can also see a typical problem in optical flow computation: the car on the right-hand side (which itself has a weak texture) is driving through the shadow of a tree. This creates a texture pattern that does not move along with the car, and hence the recovered velocity is the velocity of the street rather than the velocity of the car. In driver assistance systems and autonomous driving, the result of the optical flow computation represents an important cue, for example in crash avoidance.

7.6. Analysis operators

There have also been research directions that replace the gradient operator of the total variation with a more general analysis operator, usually in a discrete setting. Popular operators include wavelet-like transforms such as

[20] The input images are taken from the 2015 KITTI benchmark.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

curvelets (Candès, Demanet, Donoho and Ying 2006*a*, Starck, Murtagh and Fadili 2010) or shearlets (Guo, Kutyniok and Labate 2006). There have also been many attempts to learn optimal analysis operators from data; see for example Protter, Yavneh and Elad (2010), Peyré, Fadili and Starck (2010), Chen *et al.* (2014*b*) and Hawe, Kleinsteuber and Diepold (2013).

Let

$$\Phi : \mathbb{R}^{m\times n} \to \mathbb{C}^{k_1\times\cdots\times k_K}$$

be a linear transform that maps an image of size $m \times n$ pixels to a complex space of dimension $k_1 \times \cdots \times k_K$, where the dimensions k_i, $i = 1, \ldots, K$ usually depend on the number of filters, orientations, and scales used in the transform. When using the operator Φ within a regularization term, the part of the transform that computes the approximation (coarse) coefficients is usually skipped since these coefficients are generally not sparse. Hence we cannot assume that Φ is invertible. A straightforward extension of the discrete ROF model (2.6) is given by

$$\min_u \lambda\|\Phi u\|_1 + \frac{1}{2}\|u - u^\diamond\|_2^2. \tag{7.9}$$

Sometimes it is also useful to give different weights to different scales or orientations of the transform Φ, but for the sake of simplicity we will just assume a single regularization parameter. Let us show an example using the discrete shearlet transform, since it has been shown to provide optimal sparse approximations for cartoon-like images (Guo and Labate 2007, Easley, Labate and Lim 2008, Kutyniok and Lim 2011), for image inpainting. For this we consider the following formulation:

$$\min_u \|\Phi u\|_1 + \sum_{(i,j)\in\mathcal{I}} \delta_{\{u^\diamond_{i,j}\}}(u_{i,j}),$$

where

$$\mathcal{D} = \{(i,j) : 1 \le i \le m,\ 1 \le j \le n\}$$

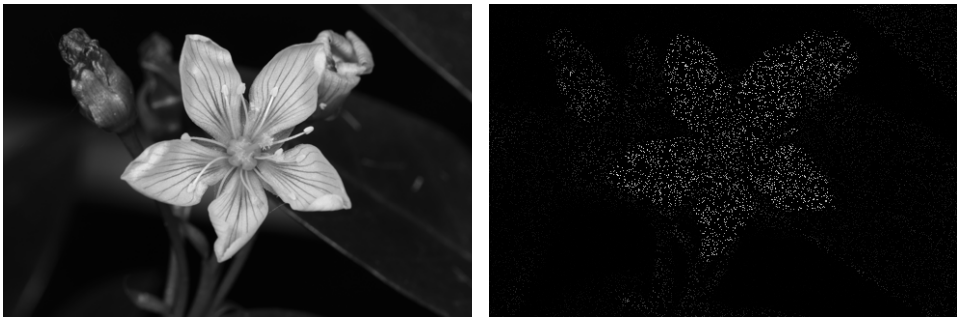
is the set of pixel indices of a discrete image of size $m \times n$, and $\mathcal{I} \subset \mathcal{D}$ is the subset of known pixels of the image $u^\diamond$. After transforming to a saddle-point problem, the solution of the inpainting problem can be computed using the PDHG algorithm. It just remains to give the proximal map with respect to the data term

$$g(u) = \sum_{(i,j)\in\mathcal{I}} \delta_{\{u^\diamond_{i,j}\}}(u_{i,j}).$$

Obviously, the solution of the proximal map is given by

$$\hat{u} = \operatorname{prox}_{\tau g}(\tilde{u}) \Leftrightarrow \hat{u}_{i,j} = \begin{cases} u^\diamond_{i,j} & \text{if } (i,j) \in \mathcal{I}, \\ \tilde{u}_{i,j} & \text{else.} \end{cases}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) original image (b) 90% missing data

(c) TV regularization (d) shearlet regularization

Figure 7.7. Image inpainting using shearlet regularization. (a) Original image, and (b) input image with a randomly chosen fraction of 10% of the image pixels. (c) Reconstruction using TV regularization, and (d) reconstruction using the shearlet model. Observe that the shearlet-based model leads to significantly better reconstruction of small-scale and elongated structures.

Figure 7.7 shows the application of the inpainting model to recover an image from only 10% known image pixels. For comparison we also show the result with TV regularization, which can be realized by replacing Φ with the discrete gradient operator D. This result shows that penalizing the ℓ_1-norm of the shearlet transform produces a significantly better approximation compared to the TV model. In particular, elongated structures are reconstructed with far more detail.

7.7. The Mumford–Shah model

The next functional we mention here is the celebrated Mumford–Shah functional (Mumford and Shah 1989). The minimization problem reads

$$\min_u \int_{\Omega\setminus S_u} |\nabla u(x)|^2 \,\mathrm{d}x + \nu \mathcal{H}^{d-1}(S_u) + \lambda \int_\Omega (u - u^\diamond)^2 \,\mathrm{d}x, \tag{7.10}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

where the image domain Ω is a subset of $\mathbb{R}^d$, $u^\diamond$ is a given image, and $u \in \mathrm{SBV}(\Omega)$ is the smooth approximation image. The space $\mathrm{SBV}(\Omega)$ refers to the so-called space of special functions of bounded variations (Ambrosio *et al.* 2000, Attouch, Buttazzo and Michaille 2014). It is a subspace of the space $\mathrm{BV}(\Omega)$ introduced in Section 7.1, and contains functions whose distributional derivatives consist only of a jump (discontinuity) part and an absolutely continuous gradient. The parameters $\nu > 0$ and $\lambda > 0$ are tuning parameters, used to balance between the different terms. $S_u \subset \Omega$ refers to the so-called jump set, that is, the set of points where the function u is allowed to jump and $\mathcal{H}^{d-1}$ is the $(d-1)$-dimensional Hausdorff measure (Ambrosio *et al.* 2000, Attouch *et al.* 2014, Evans and Gariepy 1992), which is, for $d = 2$, the length of the jump set S_u and hence the total length of edges in u. The main difference between the ROF functional and the MS functional is as follows. While the ROF functional penalizes discontinuities proportional to their jump height, the MS functional penalizes discontinuities independently of their jump height and hence allows for better discrimination between smooth and discontinuous parts of the image. We must stress that the MS functional is very hard to minimize. The reason is that the jump set S_u is not known beforehand and hence the problem becomes a non-convex optimization problem. Different numerical approaches have been proposed to find approximate solutions to the Mumford–Shah problem (Ambrosio and Tortorelli 1992, Chambolle 1999, Chan and Vese 2002, Pock *et al.* 2009).

Here we focus on the work by Alberti, Bouchitté and Dal Maso (2003), who proposed a method called the *calibration* method to characterize global minimizers of the MS functional. The approach is based on a convex representation of the MS functional in a three-dimensional space $\Omega \times \mathbb{R}$, where the third dimension is given by the value $t = u(x)$. The idea of the calibration method is to consider the maximum flux of a vector field $\varphi = (\varphi^x, \varphi^t) \in C_0(\Omega \times \mathbb{R}; \mathbb{R}^{d+1})$ through the interface of the subgraph

$$\mathbf{1}_u(x,t) = \begin{cases} 1 & \text{if } t < u(x), \\ 0 & \text{else,} \end{cases} \tag{7.11}$$

which allows us to distinguish between smooth and discontinuous areas in the function u in an elegant way. It turns out that we can find convex constraints K on the vector field such that the supremum of the flux of the vector field through the interface is identical to the value of the Mumford–Shah functional in (7.10). Formally, this energy is written as a minimal surface energy of the form

$$\mathrm{MS}(u) = \sup_{\varphi \in K} \int_{\Omega \times \mathbb{R}} \varphi D\mathbf{1}_u, \tag{7.12}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

where the convex set K is given by

$$K = \Big\{ \varphi \in C_0(\Omega \times \mathbb{R}; \mathbb{R}^{d+1}) : \quad (7.13)$$
$$\varphi^t(x,t) \geq \frac{\varphi^x(x,t)^2}{4} - \lambda(t - u^\diamond(x))^2, \left| \int_{t_1}^{t_2} \varphi^x(x,s)\,\mathrm{d}s \right| \leq \nu \Big\},$$

where the inequalities in the definition of K hold for all $x \in \Omega$ and for all $t, t_1, t_2 \in \mathbb{R}$. Observe that the set K includes non-local constraints, which is challenging from a computational point of view.

In particular, if the supremum is attained by a divergence-free vector field, the divergence theorem provides a sufficient condition of optimality for u. Such a vector field is then said to be a calibration.

Although (7.12) looks almost like a convex optimization problem, we must take into account the constraint that $\mathbf{1}_u$ is a binary function. The standard approach is to relax this constraint by replacing $\mathbf{1}_u$ with a function $v \in \mathrm{BV}(\Omega \times \mathbb{R}; [0,1])$ which satisfies

$$\lim_{t \to -\infty} v(x,t) = 1, \quad \lim_{t \to +\infty} v(x,t) = 0, \quad (7.14)$$

such that (7.12) becomes the convex problem

$$\min_v \sup_{\varphi \in K} \int_{\Omega \times \mathbb{R}} \varphi Dv. \quad (7.15)$$

Clearly, being a relaxation of the original problem, the question remains whether a minimizer of (7.15) translates to a global minimizer of the Mumford–Shah problem. In particular, this would be true if a minimizer v^* of the above optimization problem were binary, which would imply that the supremum is attained by a divergence-free vector field and hence a calibration is found. For some particular cases such as edges and triple junction, it is known that such a calibration exists (Alberti *et al.* 2003, Dal Maso, Mora and Morini 2000, Mora and Morini 2001). For other cases such as 'crack-tip', the proof of the existence of a calibration remains an unsolved problem (Bonnet and David 2001), and perhaps one might not expect to obtain a binary v^* in this situation.

Let us quickly develop a discrete version of the calibration method. We consider a spatially discrete function $v \in \mathbb{R}^{m \times n \times r}$ on a three-dimensional regular grid of $m \times n \times r$ voxels. We also associate discrete values t_k, $k = 1, \ldots, r$ of the range of the given function $u^\diamond \in \mathbb{R}^{m \times n}$ with the discrete function v. Usually the range of $u^\diamond$ is in $[0,1]$ so that $t_k = (k-1)/(r-1)$ is a natural choice.

We approximate the three-dimensional gradient operator, again using a simple finite difference operator $\mathrm{D} : \mathbb{R}^{m \times n \times r} \to \mathbb{R}^{m \times n \times r \times 3}$, which is implemented as usual, using finite differences. The operator is the extension of

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(2.4) to three dimensions, and is defined by

$$(\mathrm{D}v)_{i,j,k,1} = \begin{cases} v_{i+1,j,k} - v_{i,j,k} & \text{if } 1 \le i < m, \\ 0 & \text{else,} \end{cases}$$
$$(\mathrm{D}v)_{i,j,k,2} = \begin{cases} v_{i,j+1,k} - v_{i,j,k} & \text{if } 1 \le j < n, \\ 0 & \text{else,} \end{cases} \tag{7.16}$$
$$(\mathrm{D}v)_{i,j,k,3} = \begin{cases} v_{i,j,k+1} - v_{i,j,k} & \text{if } 1 \le k < r, \\ 0 & \text{else.} \end{cases}$$

We also introduce a discrete field variable $\mathbf{p} = (p_1, p_2, p_3) \in \mathbb{R}^{m\times n\times r\times 3}$, which can also be written in the form $\mathbf{p} = (\mathbf{p}_{1,1,1}, \ldots, \mathbf{p}_{m,n,r})$, where $\mathbf{p}_{i,j,k} = (p_{i,j,k,1}, p_{i,j,k,2}, p_{i,j,k,3}) \in \mathbb{R}^3$ is the per voxel field variable. Furthermore, we need a discrete version of the convex set K defined in (7.13):

$$\mathrm{K} = \Bigg\{ \mathbf{p} \in \mathbb{R}^{m\times n\times r\times 3} : p_{i,j,k,3} \ge \frac{p_{i,j,k,1}^2 + p_{i,j,k,2}^2}{4} - \lambda(t_k - u_{i,j}^{\diamond})^2, \\ \left| \sum_{k=k_1}^{k_2} (p_{i,j,k,1}, p_{i,j,k,2}) \right|_2 \le \nu, \text{ for all } i,j,k,\ k_1 \le k_2 \Bigg\}. \tag{7.17}$$

Finally, we define a convex set C that constrains the variable v to belong to a set of relaxed binary functions that satisfy the required boundary conditions:

$$\mathrm{C} = \left\{ v \in \mathbb{R}^{m\times n\times r} : v_{i,j,k} \in [0,1],\ v_{i,j,1} = 0,\ v_{i,j,r} = 1, \text{ for all } i,j,k \right\} \tag{7.18}$$

With this, the discrete version of (7.15) is given by the saddle-point problem

$$\min_{v} \max_{\mathbf{p}} \langle \mathrm{D}v, \mathbf{p} \rangle + \delta_{\mathrm{C}}(v) - \delta_{\mathrm{K}}(\mathbf{p}),$$

which can be solved using Algorithm 6. The critical part of the implementation of the algorithm is the solution of the projection of $\mathbf{p}$ onto K:

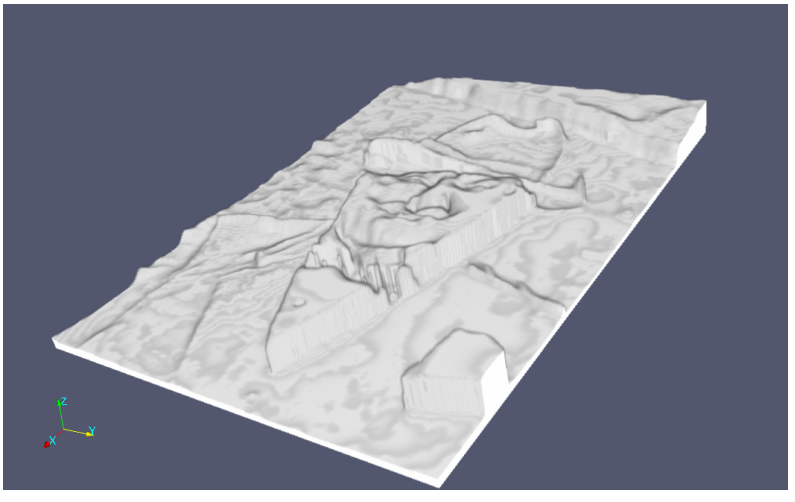
$$\hat{\mathbf{p}} = \Pi_{\mathrm{K}}(\tilde{\mathbf{p}}) = \arg\min_{\mathbf{p}\in\mathrm{K}} \frac{\|\mathbf{p} - \tilde{\mathbf{p}}\|^2}{2},$$

which is non-trivial since the set K contains a quadratic number (in fact $r(r+1)/2$) of coupled constraints. In order to solve the projection problem, we may adopt Dykstra's algorithm for computing the projection on the intersection of convex sets (Dykstra 1983). The algorithm performs a coordinate descent on the dual of the projection problem, which is defined in the product space of the constraints. In principle, the algorithm proceeds by sequentially projecting onto the single constraints. The projections onto the 2-ball constraints can be computed using projection formula (4.23). The projection to the parabola constraint can be computed by solving a cubic

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) original image $u^\diamond$

(b) piecewise smooth image u



(c) relaxed function v

Figure 7.8. Piecewise smooth approximation using the Mumford–Shah functional. (a) Original image $u^\diamond$, and (b) piecewise smooth approximation u extracted from the convex relaxation. (c) Three-dimensional rendering of the subgraph of the relaxed function v which approximates the subgraph $\mathbf{1}_u$ of the image u. Note the tendency of the Mumford–Shah functional to produce smooth regions terminated by sharp discontinuities.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

polynomial or by adopting Newton's method. Once a solution v is computed from the saddle-point problem, an approximate minimizer u can be computed for example by extracting the $\frac{1}{2}$-level set of v.

Figure 7.8 shows the result of the Mumford–Shah functional obtained from the convex representation. The solution u is composed of smooth regions terminated by sharp discontinuities. We also show a three-dimensional rendering of the relaxed function v which, if binary, is exactly the subgraph of the function u.

7.8. Convex regularization with non-convex data terms

Pock *et al.* (2008, 2010) have shown that the calibration method can also be used to compute exact solutions of a certain class of optimization problems that have important applications in imaging. The class of problems is given by

$$\min_u \int_\Omega f(x, u(x), Du), \tag{7.19}$$

where the function $f(x,t,p)$ can be non-convex in t but has to be convex in p. It turns out that a global minimizer of this problem can be computed by solving (7.15) with the following convex set of constraints:

$$K = \big\{\varphi \in C_0(\Omega \times \mathbb{R}; \mathbb{R}^{d+1}) : \varphi^t(x,t) \geq f^*(x,t,\varphi^x(x,t)),\ \forall x, t \in \Omega \times \mathbb{R}\big\}, \tag{7.20}$$

where f^* denotes the convex conjugate of $f(x,t,p)$ with respect to p. Observe that, in contrast to the convex set associated with the convex representation of the Mumford–Shah functional, this convex set is local in $\Omega \times \mathbb{R}$ and hence much more amenable to numerical optimization. The discretization of the problem is very similar to the discretization of the Mumford–Shah functional and hence we omit it.

A particularly interesting class of optimization problems is given by total variation regularization plus a quite arbitrary non-convex data term, for example given by the matching costs of a stereo problem. In this case, the convex set (7.20) completely decomposes into a set of simple and pointwise constraints. Equivalently, the problem can be solved by solving a ROF problem in three dimensions, and extracting the 0-level set of the solution, as explained later on in Section 7.9. This has the advantage that we can implement an accelerated block-coordinate descent as explained in Example 4.15. The resulting algorithm is quite efficient. It usually needs only a very small number of iterations (20–30) to give a good approximate solution. See Chambolle and Pock (2015*b*) for more information.

Figure 7.9 shows an example of computing the globally optimal solution of a total variation regularized stereo model using accelerated block descent. The non-convex data term is computed from the stereo matching costs using

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) left image (b) disparity map

Figure 7.9. Computing a globally optimal solution of a large-scale stereo problem using the calibration method. (a) Left input image showing the region around the Freiheitsplatz in the city of Graz. (b) Computed disparity map, where the intensity is proportional to the height above the ground. Black pixels indicate occluded pixels that have been determined by a left–right consistency check.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

the census transform (Zabih and Woodfill 1994). Figure 7.9(a) shows one input image of an aerial stereo image pair showing the neighbourhood of Freiheitsplatz in the city of Graz.[21] Figure 7.9(b) shows the computed disparity image obtained by solving the convexification of the non-convex stereo problem. After interchanging the left and right images we repeated the experiment. This allowed us to perform a left–right consistency check and in turn to identify occluded regions. Those pixels are shown in black.

Although the calibration method is able to compute the globally optimal solution, it is important to point out that this does not come for free. The associated optimization problem is huge because the range space of the solution also has to be discretized. In our stereo example, the disparity image is of size 1835×3637 pixels and the number of disparities was 100. This amounts to solving an optimization problem of 0.7 billion unknowns – in practice we solve the dual ROF problem, which in fact triples the number of unknowns! However, using the combination of accelerated block descent and efficient dynamic programming to solve the one-dimensional ROF problems involved allows us to solve such a huge optimization problem in 'only' 10 minutes on a 20-core machine.

Various groups have proposed extensions of these techniques to ever more difficult problems such as vector-valued data or manifold-valued data; see Lellmann and Schnörr (2011), Cremers and Strekalovskiy (2013), Goldluecke, Strekalovskiy and Cremers (2013) and Strekalovskiy, Chambolle and Cremers (2014).

7.9. Image segmentation

Image segmentation is a central problem in imaging. Let us first discuss figure-ground segmentation, whose general idea is to partition an image into two regions, one corresponding to the figure and the other to the background. A simple model consists of an energy functional that minimizes the boundary length of the segmentation plus a certain region-based segmentation criterion, for example the colour distance to given mean values of the regions. In the continuous setting, this problem can be written in the following form:

$$\min_{S \subseteq \Omega} \operatorname{Per}(S;\Omega) + \int_S w_1(x)\,\mathrm{d}x + \int_{\Omega \setminus S} w_2(x)\,\mathrm{d}x, \tag{7.21}$$

where Ω is the image domain, $\operatorname{Per}(S;\Omega)$ denotes the perimeter of the set S in Ω, and $w_{1,2} : \Omega \to \mathbb{R}^+$ are given non-negative potential functions. This problem belongs to a general class of minimal surface problems that have been studied for a long time (see for instance the monograph by Giusti 1984).

[21] Data courtesy of the Vermessungsamt Graz.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

The discrete version of this energy is commonly known as the 'Ising' model, which represents the interactions between spins in an atomic lattice and exhibits phase transitions. In computer science, the same kind of energy has been used to model many segmentation and classification tasks, and has received a lot of attention since it was understood that it could be efficiently minimized if represented as a minimum $s-t$ cut problem (Picard and Ratliff 1975) in an oriented graph $(\mathcal{V},\mathcal{E})$. Here, $\mathcal{V}$ denotes a set of vertices and $\mathcal{E}$ denotes the set of edges connecting some of these vertices. Given two particular vertices, the 'source' s and the 'sink' t, the $s-t$ minimum cut problem consists in finding two disjoint sets $S \ni s$ and $T \ni t$ with $S \cup T = \mathcal{V}$ and the cost of the 'cut' $C(S,T) = \{(u,v) \in \mathcal{E} : u \in S, v \in T\}$ is minimized. The cost of the cut can be determined by simply counting the number of edges, or by summing a certain weight w_{uv} associated with each edge $(u,v) \in \mathcal{E}$. By the Ford–Fulkerson min-cut/max-flow duality theorem (see Ahuja, Magnanti and Orlin 1993 for a fairly complete textbook on these topics), this minimal $s-t$ cut can be computed by finding a maximal flow through the oriented graph, which can be solved by a polynomial-time algorithm. In fact, there is a 'hidden' convexity in the problem. We will describe this briefly in the continuous setting; for discrete approaches to image segmentation we refer to Boykov and Kolmogorov (2004), and the vast subsequent literature. The min-cut/max-flow duality in the continuous setting and the analogy with minimal surfaces type problems were first investigated by Strang (1983) (see also Strang 2010).

We mentioned in the previous section that the total variation (7.2) is also well defined for characteristic functions of sets, and measures the length of the boundary (in the domain). This is, in fact, the 'correct' way to define the perimeter of a measurable set, introduced by R. Caccioppoli in the early 1950s. Ignoring constants, we can replace (7.21) with the following equivalent variational problem:

$$\min_{S \subseteq \Omega} \int_\Omega |D\mathbf{1}_S| + \int_\Omega \mathbf{1}_S(x) w(x)\,\mathrm{d}x, \tag{7.22}$$

where for notational simplicity we have set $w = w_1 - w_2$, and $\mathbf{1}_S$ is the characteristic function associated with the set S, that is,

$$\mathbf{1}_S(x) = \begin{cases} 1 & \text{if } x \in S, \\ 0 & \text{else.} \end{cases}$$

The idea is now to replace the binary function $\mathbf{1}_S : \Omega \to \{0,1\}$ with a continuous function $u : \Omega \to [0,1]$ such that the problem becomes convex:

$$\min_u \int_\Omega |Du| + \int_\Omega u(x) w(x)\,\mathrm{d}x, \quad \text{such that } u(x) \in [0,1] \text{ a.e. in } \Omega, \tag{7.23}$$

It turns out that the relaxed formulation is exact in the sense that any

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

thresholded solution $v = \mathbf{1}_{\{u \geq s\}}$ of the relaxed problem for any $s \in (0, 1]$ is also a global minimizer of the binary problem (Chan, Esedoḡlu and Nikolova 2006, Chambolle 2005, Chambolle and Darbon 2009). This is a consequence of the co-area formula (Federer 1969, Giusti 1984, Ambrosio *et al.* 2000), which shows that minimizing the total variation of u decomposes into independent problems on all level sets of the function u.

Interestingly, there is also a close relationship between the segmentation model (7.23) and the ROF model (7.1). In fact a minimizer of (7.23) is obtained by minimizing (7.1), with $u^\diamond = w$ being the input image, and then thresholding the solution u at the 0 level (Chambolle 2004*a*, 2005). Conversely, this relationship has also been successfully used to derive efficient combinatorial algorithms, based on parametric maximal flow approaches (Gallo, Grigoriadis and Tarjan 1989), to solve the fully discrete ROF model exactly in polynomial time (Hochbaum 2001, Darbon and Sigelle 2004, Darbon and Sigelle 2006*a*, Darbon and Sigelle 2006*b*, Chambolle and Darbon 2012), where the total variation is approximated by sum of pairwise interactions $|u_i - u_j|$.

Exploiting the relation between the ROF model and the two-label segmentation model, we can easily solve the segmentation problem by considering a discrete version of the ROF model. In our setting here, we consider a discrete image $u \in \mathbb{R}^{m \times n}$ and a discrete weighting function $w \in \mathbb{R}^{m \times n}$. The discrete model we need to solve is

$$\min_u \|\mathrm{D}u\|_{2,1} + \frac{1}{2}\|u - w\|^2.$$

It can be solved by using either Algorithm 8 or Algorithm 5 (applied to the dual problem). Let u^* denote the minimizer of the ROF problem. The final discrete and binary segmentation $\mathbf{1}_\mathcal{S}$ is given by thresholding u^* at zero:

$$(\mathbf{1}_\mathcal{S})_{i,j} = \begin{cases} 0 & \text{if } u^*_{i,j} < 0, \\ 1 & \text{else.} \end{cases}$$

Figure 7.10 shows an example of foreground–background segmentation using this approach. The weighting function w was computed using the negative log-ratio of two Gaussian mixture models (GMMs) that were fitted to the desired foreground and background regions provided by the input of a user. Let

$$\mathcal{N}(\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right)}{\sqrt{(2\pi)^d \det \boldsymbol{\Sigma}}}$$

be the (d-dimensional) normal distribution with expectation $\boldsymbol{\mu} \in \mathbb{R}^d$ and

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) input image with user input

(b) weight function

(c) segmentation

(d) background removal

Figure 7.10. Interactive image segmentation using the continuous two-label image segmentation model. (a) Input image overlaid with the initial segmentation provided by the user. (b) The weighting function w, computed using the negative log-ratio of two Gaussian mixture models fitted to the initial segments. (c) Binary solution of the segmentation problem, and (d) the result of performing background removal.

covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{d\times d}$. We let

$$G(\mathbf{x}; \boldsymbol{\mu}, \mathbf{\Sigma}, \boldsymbol{\alpha}) = \sum_{l=1}^{L} \alpha_l \mathcal{N}(\mathbf{x}, \boldsymbol{\mu}_l, \mathbf{\Sigma}_l) \tag{7.24}$$

denote a GMM with L components, where

$$\boldsymbol{\alpha} = (\alpha_l)_{l=1}^{L} \in [0,1]^L, \quad \sum_{l=1}^{L} \alpha_l = 1$$

are the mixture coefficients, $\boldsymbol{\mu} = (\boldsymbol{\mu}_l)_{l=1}^{L}$ are the means, and $\mathbf{\Sigma} = (\mathbf{\Sigma}_l)_{l=1}^{L}$ are the covariances of the Gaussian probability densities. Further, we let $G^f(\cdot; \boldsymbol{\mu}^f, \mathbf{\Sigma}^f, \boldsymbol{\alpha}^f)$ denote the Gaussian mixture model of the figure and $G^b(\cdot; \boldsymbol{\mu}^b, \mathbf{\Sigma}^b, \boldsymbol{\alpha}^b)$ the mixture model of the background.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Following the well-known GrabCut algorithm (Rother, Kolmogorov and Blake 2004), the weighting function w is given by the negative log-ratio

$$w_{i,j} = -\log\left(\frac{G^f(\mathbf{u}^\diamond_{i,j}; \boldsymbol{\mu}^f, \boldsymbol{\Sigma}^f, \boldsymbol{\alpha}^f)}{G^b(\mathbf{u}^\diamond_{i,j}; \boldsymbol{\mu}^b, \boldsymbol{\Sigma}^b, \boldsymbol{\alpha}^b)}\right),$$

where $\mathbf{u}^\diamond \in \mathbb{R}^{m\times n\times 3}$ is a given colour image. The weighting function is larger than zero if the pixel $\mathbf{u}^\diamond_{i,j}$ is more likely to be a background pixel, and smaller than zero if a pixel is more likely to be a foreground pixel. In our experiment we used Gaussian mixture models with 10 components. The Gaussian mixture models can be computed using the classical EM (expectation–maximization) algorithm. In practice, it consists in alternating the following steps, until convergence.

- Compute at each pixel of the (current) foreground/background the membership probabilities for each Gaussian,

$$\pi^f_{i,j,l} = \frac{\alpha^f_l \mathcal{N}(\mathbf{u}^\diamond_{i,j}, \boldsymbol{\mu}^f_l, \boldsymbol{\Sigma}^f_l)}{\sum_{l'=1}^k \alpha^f_l \mathcal{N}(\mathbf{u}^\diamond_{i,j}, \boldsymbol{\mu}^f_{l'}, \boldsymbol{\Sigma}^f_{l'})}$$

  for $l = 1, \dots, L$ and each foreground pixel $(i, j) \in fg$, and similarly for $(\pi^b)_{i,j,l}$, $(i, j) \in bg$ (here $fg, bg \subset \{1, \dots, n\} \times \{1, \dots, m\}$ denote the set of foreground and background pixels, respectively).

- Update the parameters:

$$\alpha^f_l = \frac{1}{\#fg} \sum_{(i,j)\in fg} \pi^f_{i,j,l}, \quad \boldsymbol{\mu}^f_l = \frac{\sum_{i,j\in fg} \mathbf{u}^\diamond_{i,j} \pi^f_{i,j,l}}{\sum_{i,j\in fg} \pi^f_{i,j,l}},$$
$$\boldsymbol{\Sigma}^f_l = \frac{\sum_{i,j\in fg} (\mathbf{u}^\diamond_{i,j} - \boldsymbol{\mu}^f_l)(\mathbf{u}^\diamond_{i,j} - \boldsymbol{\mu}^f_l)^T \pi^f_{i,j,l}}{\sum_{i,j\in fg} \pi^f_{i,j,l}},$$

  and similarly for the background.

After solving the segmentation problem, the Gaussian mixture models can be re-computed and the segmentation can be refined.

7.10. Extension to multilabel segmentation

We now describe an extension of the two-region segmentation model to multiple regions. Here the idea is to partition the image domain into a set of K disjoint image segments Ω_k, $k = 1, \dots, K$. In the continuous setting

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

such a model is given by

$$\min_{(\Omega_k)_{k=1}^K} \frac{1}{2} \sum_{k=1}^K \mathrm{Per}(\Omega_k) + \int_{\Omega_k} w_k(x)\,\mathrm{d}x, \tag{7.25}$$

$$\text{such that } \Omega = \bigcup_{k=1}^K \Omega_k,\ \Omega_k \cap \Omega_l = \emptyset, \quad \text{for all } k \neq l. \tag{7.26}$$

This model can be interpreted as the continuous version of the 'Potts' model that has also been proposed in statistical mechanics to model the interactions of spins on a crystalline lattice. It is also widely used as a smoothness term in graphical models for computer vision, and can be minimized (approximately) by specialized combinatorial optimization algorithms such as those proposed by Boykov *et al.* (2001).

The continuous Potts model (7.25) is also closely related to the seminal Mumford–Shah model (Mumford and Shah 1989), where the smooth approximation of the image is restricted to piecewise constant regions. See Chan and Vese (2001), for example, where the problem is approached using a level set method.

In the discrete setting it is known that the Potts model is NP-hard, so we cannot hope to solve this problem exactly in the continuous setting either. The most standard approach to this problem is to consider a convex relaxation similar to (7.23) by introducing a vectorial function $u = (u_1, \dots, u_K) : \Omega \to [0,1]^K$ with the constraint that $\sum_{k=1}^K u_k(x) = 1$ a.e. in Ω, and consider some vectorial version of the total variation, which coincides with half of the sum of the perimeters of the sets Ω_k if the function u is binary. A convex relaxation of (7.25) is now given by

$$\min_u \int_\Omega |Du|_{\mathcal{P}} + \sum_{k=1}^K \int_\Omega u_k(x) w_k(x)\,\mathrm{d}x, \tag{7.27}$$

$$\text{such that } u(x) \in \mathcal{S}_{K-1} \text{ for a.e. } x \in \Omega, \tag{7.28}$$

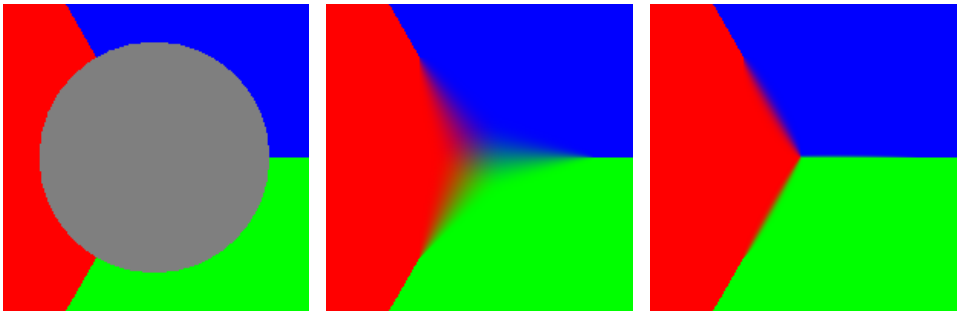
where

$$\mathcal{S}_{K-1} = \left\{ x \in \mathbb{R}_+^K : \sum_{i=1}^K x_i = 1 \right\}$$

denotes the $(K-1)$-dimensional unit simplex, and the vectorial total variation is given by

$$\int_\Omega |Du|_{\mathcal{P}} = \sup \Big\{ - \int_\Omega u(x) \cdot \operatorname{div} \varphi(x)\,\mathrm{d}x : \varphi \in C^\infty(\Omega; \mathbb{R}^{d \times K}),\ \varphi(x) \in C_{\mathcal{P}},\ \text{for all } x \in \Omega \Big\},$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) input image (b) using $C_{\mathcal{P}_1}$ (c) using $C_{\mathcal{P}_2}$

Figure 7.11. Demonstration of the quality using different relaxations. (a) Input image, where the task is to compute a partition of the grey zone in the middle of the image using the three colours as boundary constraints. (b) Colour-coded solution using the simple relaxation $C_{\mathcal{P}_1}$, and (c) result using the stronger relaxation $C_{\mathcal{P}_2}$. Observe that the stronger relaxation exactly recovers the true solution, which is a triple junction.

where $C_{\mathcal{P}}$ is a convex set, for which various choices can be made. If the convex set is given by

$$C_{\mathcal{P}_1} = \left\{ \xi = (\xi_1, \dots, \xi_K) \in \mathbb{R}^{d \times K} : |\xi_k|_2 \leq \frac{1}{2}, \text{ for all } k \right\},$$

the vectorial total variation is simply the sum of the total variations of the single channels (Zach, Gallup, Frahm and Niethammer 2008). Chambolle, Cremers and Pock (2012) have shown that a strictly larger convex function is obtained by means of the so-called paired calibration (Lawlor and Morgan 1994, Brakke 1995). In this case, the convex set is given by

$$C_{\mathcal{P}_2} = \left\{ \xi = (\xi_1, \dots, \xi_K) \in \mathbb{R}^{d \times K} : |\xi_k - \xi_l|_2 \leq 1, \text{ for all } k \neq l \right\},$$

which has a more complicated structure than $C_{\mathcal{P}_1}$ but improves the convex relaxation. See Figure 7.11 for a comparison. Note that unlike in the two-phase case, the relaxation is not exact. Thresholding or rounding a minimizer of the relaxed problem will only provide an approximate solution to the problem (Lellmann, Lenzen and Schnörr 2013).

For the numerical implementation we consider a straightforward discretization similar to the previous models. We consider a discrete labelling function $\mathbf{u} = (u_1, \dots, u_K) \in \mathbb{R}^{m \times n \times K}$, and we consider the usual finite difference approximation of the gradient operator $\mathbf{D} : \mathbb{R}^{m \times n \times K} \to \mathbb{R}^{m \times n \times K \times 2}$, where $\mathbf{Du} = (\mathrm{D}u_1, \dots, \mathrm{D}u_K)$ and D is defined in (2.4). Furthermore, we are given a discrete weight function $\mathbf{w} = (w_1, \dots, w_K) \in \mathbb{R}^{m \times n \times K}$. Therefore

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

the discrete pendant of the Potts model is given by

$$\min_{\mathbf{u}} \|\mathbf{Du}\|_{2,\mathcal{P}} + \sum_{k=1}^{K} \langle u_k, w_k \rangle, \quad \text{such that } \mathbf{u}_{i,j} \in \mathcal{S}_{K-1},$$

and the vectorial total variation that is intended to measure half the length of the total boundaries is given by

$$\|\mathbf{Du}\|_{2,\mathcal{P}} = \sup_{\mathbf{P}} \langle \mathbf{Du}, \mathbf{P} \rangle, \quad \text{such that } \mathbf{P}_{i,j} \in C_{\mathcal{P}} \text{ for all } i, j,$$

where $\mathbf{P} \in \mathbb{R}^{m \times n \times 2 \times K}$ is the tensor-valued dual variable. Combining the two last equations already leads to a saddle-point problem that can be solved using Algorithm 6 (PDHG). It remains to detail the pixelwise projections onto the simplex constraints $\mathbf{u}_{i,j} \in \mathcal{S}_{K-1}$ and the constraints $\mathbf{P}_{i,j} \in C_{\mathcal{P}}$ on the dual variables. The projection on the $(K-1)$-dimensional simplex $\mathcal{S}_{K-1}$ can be done for each pixel independently in $K \log K$ time or even in expected linear time; see for example Duchi, Shalev-Shwartz, Singer and Chandra (2008). The complexity of the projection onto $C_{\mathcal{P}}$ depends on the particular choice of the set. If we choose the weaker set $C_{\mathcal{P}_1}$ the projection reduces to K independent projections onto the 2-ball with radius $1/2$. If we choose the stronger relaxation $C_{\mathcal{P}_2}$, no closed-form solution is available to compute the projection. A natural approach is to implement Dykstra's iterative projection method (Dykstra 1983), as $C_{\mathcal{P}_2}$ is the intersection of simple convex sets on which a projection is straightforward. Another efficient possibility would be to introduce Lagrange multipliers for the constraints defining this set, but in a progressive way as they get violated. Indeed, in practice, it turns out that few of these constraints are actually active, in general no more than two or three, and only in a neighbourhood of the boundary of the segmentation.

Figure 7.12 shows the application of interactive multilabel image segmentation using four phases. We again use the user input to specify the desired regions, and we fit Gaussian mixture models (7.24) $G^k(\cdot; \boldsymbol{\mu}^k, \boldsymbol{\Sigma}^k, \boldsymbol{\alpha}^k))$, $k = 1, \dots, K$ with 10 components to those initial regions. The weight functions w_k are computed using the negative log probability of the respective mixture models, that is,

$$w_{i,j,k} = -\log G^k(\mathbf{u}^{\diamond}_{i,j}; \boldsymbol{\mu}^k, \boldsymbol{\Sigma}^k, \boldsymbol{\alpha}^k), \quad k = 1, \dots, K.$$

It can be observed that the computed phases u_k are almost binary, which indicates that the computed solution is close to a globally optimal solution.

7.11. Curvature

Using curvature information in imaging is mainly motivated by findings in psychology that so-called subjective (missing) object boundaries that

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) input image with user input

(b) segmentation

(c) phase 1

(d) phase 2

(e) phase 3

(f) phase 4

Figure 7.12. Interactive image segmentation using the multilabel Potts model. (a) Input image overlaid with the initial segmentation provided by the user. (b) Final segmentation, where the colour values correspond to the average colours of the segments. (c–f) The corresponding phases u_k. Observe that the phases are close to binary and hence the algorithm was able to find an almost optimal solution.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

are seen by humans are linear or curvilinear (Kanizsa 1979). Hence such boundaries can be well recovered by minimizing the 'elastica functional'

$$\int_\gamma (\alpha + \beta\kappa^2)\,\mathrm{d}\gamma, \tag{7.29}$$

where $\alpha > 0$, $\beta > 0$ are weighting parameters, γ is a smooth curve, and κ is its curvature. Psychological experiments also suggest that such a process must take place at a very early stage in the human visual system, as it is so strong that it cannot be resolved even if the structures seen become absurd to the human observer. It is therefore natural that curvature information provides a very strong prior for recovering missing parts of an image (Masnou and Morel 2006) or to resolve occluded objects (Nitzberg, Mumford and Shiota 1993).

In order to construct a regularization term for images, the most natural idea (Masnou and Morel 2006, Ambrosio and Masnou 2003) is to apply the elastica energy to the level lines of a sufficiently smooth (twice-differentiable) image:

$$\int_\Omega |\nabla u| \left(\alpha + \beta \left(\operatorname{div} \frac{\nabla u}{|\nabla u|}\right)^2\right) \mathrm{d}x. \tag{7.30}$$

Here, $\operatorname{div}(\nabla u/|\nabla u|) = \kappa_{\{u=u(x)\}}(x)$ represents the curvature of the level line/surface of u passing through x, and thanks to the co-area formula this expression can (or should) also be written as

$$\int_{-\infty}^{+\infty} \int_{\Omega\cap\partial\{u>s\}} (\alpha + \beta\kappa^2_{\{u>s\}})\,\mathrm{d}\mathcal{H}^{d-1}(x)\,\mathrm{d}s.$$

Masnou and Morel (2006) propose an algorithm based on dynamic programming that can find globally optimal solutions to this highly non-convex optimization problem. The authors apply the elastica model to the problem of recovering missing parts of an image. Although the model is simple, it yields faithful reconstructions.

However, curvature is notoriously difficult to minimize and hence its utilization as a regularization term in imaging is still limited. In Bredies, Pock and Wirth (2013), a convex relaxation approach was proposed to find a convex representation of curvature-dependent regularization functionals. The idea is based on work by Citti and Sarti (2006), who proposed representing images in the so-called roto-translation space in order to perform amodal completion. The basic idea is to lift the gradient of a two-dimensional image to a three-dimensional space, where the third dimension is given by the orientation of the image gradient. The lifting of the image gradient is defined via the non-negative measure $\mu = \mu(Du)$ given by

$$\int_\Omega \varphi\,\mathrm{d}\mu = \int_\Omega \varphi(x, -\sigma(x)^\perp)\,\mathrm{d}|Du|,$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

for all test functions that are continuous and compactly supported on the image domain Ω. Here $\sigma(x)$ denotes the local orientation of the image gradient (*e.g.* $Du = \sigma|Du|$), and $\perp$ denotes an anticlockwise rotation by 90 degrees, and hence $-\sigma(x)^{\perp}$ is such that the image function u is increasing on its left-hand side. An appealing property of this representation is that the image gradient can be recovered from the lifted representation via a linear constraint:

$$\int_{\Omega} \varphi \cdot \mathrm{d}Du = \int_{\Omega \times \mathbb{S}^1} \varphi(x) \cdot \vartheta^{\perp} \, \mathrm{d}\mu(x, \vartheta).$$

In the lifted space a new regularization term can be defined that penalizes curvature information. Such a regularizer – called total vertex regularization (TVX) – is given by

$$\sup_{\psi(x,\cdot) \in B_\rho} \int_{\Omega \times \mathbb{S}^1} D_x \psi(x, \vartheta) \cdot \vartheta \, \mathrm{d}\mu(x, \vartheta), \tag{7.31}$$

where $D_x \psi(x, \vartheta) \cdot \vartheta$ is the directional derivative in the direction of θ, and B_ρ defines the pre-dual ball for any metric ρ on $\mathbb{S}^1$:

$$B_\rho = \{\varphi \in C(\mathbb{S}^1) : \varphi(\eta_1) - \varphi(\eta_2) \leq \rho(\eta_1, \eta_2) \quad \text{for all } (\eta_1, \eta_2) \in \mathbb{S}^1 \times \mathbb{S}^1\} \tag{7.32}$$

Several metrics ρ can be considered: One choice is the ℓ_0-metric or Potts metric, which counts the number of 'corners'. In this case, $\rho(\eta_1, \eta_2) = 1$ for $\eta_1 \neq \eta_2$, and the pre-dual ball is simply given by

$$B_0 = \mathbf{1} \cdot \mathbb{R} + \left\{ \|\varphi\|_\infty \leq \frac{1}{2} \right\}, \tag{7.33}$$

which is the set of all constant functions plus the ∞-ball of radius $1/2$. In what follows, we call the corresponding model the TVX^0 model.

Another natural choice is the ℓ_1-metric, which is equivalent to the total absolute curvature on smooth parts and penalizes the unsigned external angle between two joining line segments: hence $\rho(\eta_1, \eta_2) = \mathrm{dist}(\eta_1, \eta_2)$. This model is referred to as the TVX^1 model. The corresponding pre-dual ball B_1 is given by

$$B_1 = \{\varphi \in C(\mathbb{S}^1) : \|D\varphi\|_\infty \leq 1\}, \tag{7.34}$$

that is, the set of 1-Lipschitz continuous functions.

An extension to non-metrics on $\mathbb{S}^1$ (*e.g.* the squared curvature) requires lifting to the space of pairs of orientations; see for example Bredies, Pock and Wirth (2015*b*), Schoenemann and Cremers (2007) and Schoenemann, Masnou and Cremers (2011).

Interestingly, the total variation in the lifted space is equivalent to the norm $\|\mu\|_{\mathcal{M}}$, which is the total variation or mass of the measure μ, defined

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

in the classical sense by

$$\|\mu\|_{\mathcal{M}} := \sup\left\{\sum_i |\mu(E_i)| : E_i \text{ disjoint sets}\right\},$$

and which generalizes the L^1-norm to measures (Evans and Gariepy 1992). This enforces sparsity of the lifted measure μ. In practice, it turns out that a combination of total variation regularization and total vertex regularization performs best. An image restoration model combining both total variation and total vertex regularization is given by

$$\min_{(u,\mu)} \alpha \sup_{\psi(x,\cdot)\in B_\rho} \int_{\Omega\times\mathbb{S}^1} D_x\psi(x,\vartheta)\cdot\vartheta \,\mathrm{d}\mu(x,\vartheta) + \beta\|\mu\|_{\mathcal{M}} + \frac{1}{2}\|u-u^\diamond\|^2,$$
$$\text{such that } (u,\mu)\in L^\mu_{Du} = \{(u,\mu) \mid \mu \text{ is the lifting of } Du\}, \tag{7.35}$$

where α and β are tuning parameters. Clearly, the constraint that μ is a lifting of Du represents a non-convex constraint. A convex relaxation of this constraint is obtained by replacing L^μ_{Du} with the following convex constraint:

$$L^\mu_{Du} = \left\{(u,\mu) \mid \mu\geq 0,\ \int_\Omega \varphi\cdot \,\mathrm{d}Du = \int_{\Omega\times\mathbb{S}^1} \varphi(x)\cdot\vartheta^\perp \,\mathrm{d}\mu(x,\vartheta)\right\}, \tag{7.36}$$

for all smooth test functions φ that are compactly supported on Ω. With this, the problem becomes convex and can be solved. However, it remains unclear how close minimizers of the relaxed problem are to minimizers of the original problem.

It turns out that the total vertex regularization functional works best for inpainting tasks, since it tries to connect level lines in the image with curves with a small number of corners or small curvature. Tackling the TVX models numerically is not an easy task because the lifted measure is expected to concentrate on line-like structures in the roto-translational space. Let us assume our image is defined on a rectangular domain $\Omega = [0,n)\times[0,m)$. On this domain we consider a collection of square pixels $\{\Omega_{i,j}\}_{i=1,j=1}^{m,n}$ with $\Omega_{i,j} = [j-1,j)\times[i-1,i)$, such that $\Omega = \bigcup_{i=1,j=1}^{m,n}\Omega_{i,j}$. Furthermore, we consider a collection of grid points $\{x_{i,j}\}_{i=1,j=1}^{m,n}$ with $x_{i,j} = (j,i)$ such that the grid points $x_{i,j}$ are located on the lower right corners of the corresponding image pixels $\Omega_{i,j}$. Using the collection of image pixels, we consider a piecewise constant image

$$u = \{u:\Omega\to\mathbb{R} : u(x) = U_{i,j}, \text{ for all } x\in\Omega_{i,j}\},$$

where $U\in\mathbb{R}^{m\times n}$ is the discrete version of the continuous image u.

Following Bredies *et al.* (2015*b*), we use a neighbourhood system based on a set of o distinct displacement vectors $\delta_k = (\delta_k^1,\delta_k^2)\in\mathbb{Z}^2$. On a regular grid, it is natural to define a system consisting of $4, 8, 16, 32$, *etc.* neighbours.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

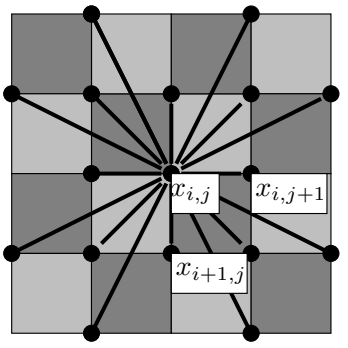


Figure 7.13. A 16-neighbourhood system on the grid. The black dots refer to the grid points $x_{i,j}$, the shaded squares represent the image pixels $\Omega_{i,j}$, and the line segments $l_{i,j,k}$ connecting the grid points are depicted by thick lines.

Figure 7.13 depicts an example based on a neighbourhood system of 16 neighbours. The displacement vectors naturally imply orientations $\vartheta_k \in \mathbb{S}^1$, defined by $\vartheta_k = \delta_k/|\delta_k|_2$. We shall assume that the displacement vectors δ_k are ordered such that the corresponding orientations ϑ_k are ordered on $\mathbb{S}^1$.

Next, we consider a collection of line segments $\{l_{i,j,k}\}_{i=1,j=1,k=1}^{m,n,o}$, where the line segments $l_{i,j,k} = [x_{i,j}, x_{i,j} + \delta_k]$ connect the grid points $x_{i,j}$ to a collection of neighbouring grid points $x_{\hat{i},\hat{j}}$, as defined by the neighbourhood system. The measure μ is discretized on these line segments: we assume it is given by

$$\mu = \sum_{i,j,k} V_{i,j,k} \mathcal{H}^1|_{l_{i,j,k}} \otimes \delta_{\theta_k}.$$

It means that μ is the finite-dimensional combination of small 'line measures' (here $\mathcal{H}^1$ is the Hausdorff one-dimensional measure) supported by the segment $l_{i,j,k}$ for $\theta = \theta_k$. The coordinates $V \in \mathbb{R}^{m\times n\times o}$ can be recovered from μ by computing the averages along the line segment $l_{i,j,k}$ for all i, j, k:

$$V_{i,j,k} = \frac{1}{|\delta_k|_2} \int_{l_{i,j,k}} \mathrm{d}\mu(x, \vartheta_k).$$

In order to approximate the compatibility constraint in (7.36), we introduce a family of test functions $\varphi_{i,j}(x) : \Omega \to \mathbb{R}^2$ which are given by

$$\varphi_{i,j}(x) = \begin{pmatrix} \max\{0, 1 - |x_1 - j|\}\mathbf{1}_{[i-1,i]}(x_2) \\ \mathbf{1}_{[j-1,j]}(x_1) \max\{0, 1 - |x_2 - i|\} \end{pmatrix}.$$

Observe that the test functions are given by the product of a triangular and a rectangular interpolation kernel, which can be interpreted as performing a linear interpolation in one direction and a nearest-neighbour interpolation in the other direction. Using these test functions in (7.36), and thanks to

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

the fact that the image u is piecewise constant, we obtain the collection of discrete constraints

$$\begin{aligned}\begin{pmatrix} U_{i,j+1} - U_{i,j} \\ U_{i+1,j} - U_{i,j} \end{pmatrix} &= \int_{[j-1,j+1]\times[i-1,i+1]\times\mathbb{S}^1} \phi_{i,j}(x) \cdot \vartheta^\perp \,\mathrm{d}\mu(x,\theta) \\ &= \sum_{\hat{\imath},\hat{\jmath}} \sum_{k=1}^{o} V_{\hat{\imath},\hat{\jmath},k} \int_{l_{\hat{\imath},\hat{\jmath},k}} \vartheta_k^\perp \cdot \varphi_{i,j}(x) \,\mathrm{d}x \quad \Longleftrightarrow \quad \mathrm{D}U = \mathrm{C}V.\end{aligned}$$

The operator $\mathrm{D} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n\times 2}$ is the usual finite differences operator, as introduced in (2.4), and the operator $\mathrm{C} : \mathbb{R}^{m\times n\times o} \to \mathbb{R}^{m\times n\times 2}$ holds the coefficients given by the value of the line integrals. Note that C will be very sparse since for each grid point $x_{i,j}$ only a small number of line segments will intersect with the non-zero part of the test function $\varphi_{i,j}$. Observe that the integrals over the line segments can be computed in closed form since the test functions $\varphi_{i,j}$ are piecewise affine functions.

In the lifted space, we will need to compute the directional derivatives $D_x\psi(x,\vartheta)\cdot\vartheta$. We assume that ψ is continuous and linear on the line segments, and hence we only need to store ψ on the end-points of the line segments $l_{i,j,k}$. For this we use a discrete test function $P \in \mathbb{R}^{m\times n\times o}$, which holds the values of ψ on the end-points of the corresponding line segments. In turn, the directional derivatives can be computed by simple finite differences along the line segments. We therefore introduce a linear operator $\mathrm{A} : \mathbb{R}^{m\times n\times o} \to \mathbb{R}^{m\times n\times o}$, which is defined by

$$(\mathrm{A}P)_{i,j,k} = \frac{P_{i^+,j^+,k} - P_{i,j,k}}{|\delta_k|_2},$$

where $i^+ = \max\{1, \min\{m, i + \delta_k^2\}\}$, and $j^+ = \max\{1, \min\{n, j + \delta_k^1\}\}$.

In order to implement the TVX^1 model, we further introduce a linear operator $\mathrm{B} : \mathbb{R}^{m\times n\times o} \to \mathbb{R}^{m\times n\times o}$, which computes the angular derivative of the test function ψ. Assuming again that ψ is linear in the angular direction between the grid points $x_{i,j}$, its angular derivative can again be computed by a finite differences operator:

$$(\mathrm{B}P)_{i,j,k} = \begin{cases} \dfrac{P_{i,j,k+1} - P_{i,j,k}}{\alpha_k} & \text{if } k < o, \\ \dfrac{P_{i,j,1} - P_{i,j,k}}{\alpha_k} & \text{if } k = o, \end{cases}$$

where the factors α_k are given by angular differences between the two corresponding line segments.

As mentioned above, the total variation in the lifted space is given by the norm $\|\mu\|_{\mathcal{M}}$. The discrete variant using the discrete measure V reads

$$\sum_{i,j,k} |\delta_k|_2 V_{i,j,k},$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) Input image F

(b) TVX^0, $\lambda = 1/2$ (c) TVX^0, $\lambda = 1/4$ (d) TVX^0, $\lambda = 1/8$ (e) TVX^0, $\lambda = 1/16$

(f) TVX^1, $\lambda = 1/2$ (g) TVX^1, $\lambda = 1/4$ (h) TVX^1, $\lambda = 1/8$ (i) TVX^1, $\lambda = 1/16$

Figure 7.14. Comparison of TVX^0 (b–e) and TVX^1 (f–i) regularization for shape denoising. One can see that TVX^0 leads to a gradually simplified polygonal approximation of the shape in F whereas TVX^1 leads to an approximation by piecewise smooth shapes.

where $|\delta_k|_2$ is the length of the corresponding line segment.

We are now ready to state the discrete approximations of the TVX models:

$$\min_{U,V} \max_{P} \ \alpha \sum_{i,j,k} |\delta_k|_2 V_{i,j,k} + \beta \sum_{i,j,k} |\delta_k|_2 V_{i,j,k} (\mathrm{A}P)_{i,j,k} + g(U),$$

$$\text{such that } V_{i,j,k} \geq 0, P_{i,j,k} \in B_\rho, \ \mathrm{D}U = \mathrm{C}V.$$

The function $g(U)$ is a data-fitting term, which defines the type of application of the model. The TVX^0 model is obtained by using the constraint $|P_{i,j,k}| \leq 1/2$ and the TVX^1 model is obtained by using the constraint $|(\mathrm{B}P)_{i,j,k}| \leq 1$.

Introducing additional Lagrange multipliers, the above problems can be easily turned into a standard saddle-point form which can be tackled by Algorithm 6 (PDHG). We leave this task to the reader.

In a first experiment, we apply the TVX models to shape denoising. Consider a shape encoded as a binary image F. The idea is to compute

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

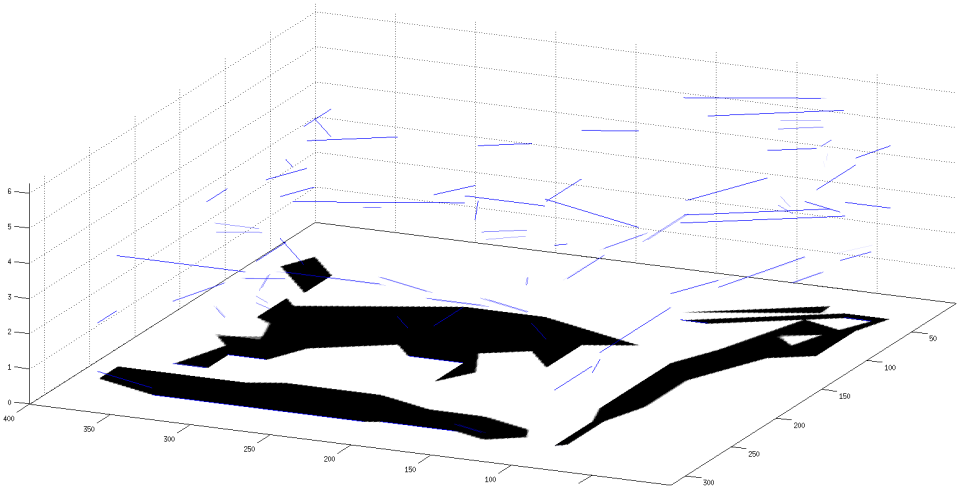


Figure 7.15. Visualization of the measure μ in the roto-translation space for the image of Figure 7.14(e), obtained using TVX^0 regularization. Observe that the measure μ indeed concentrates on thin lines in this space.

a (relaxed) binary image U that approximates the shape of F by minimizing the TVX energy. For this, we consider a data-fitting term $g(U) = \lambda\langle U, 0.5 - F\rangle + \delta_{[0,1]^{m\times n\times o}}(U)$, which measures the difference between the shapes in U and F and forces the values of U to stay in the interval $[0, 1]$. In all experiments we set $\alpha = 0.01$, $\beta = 1$ and we use $o = 32$ discrete orientations. In Figure 7.14 we show the results of TVX^1 and TVX^0 regularization using different weights λ in the data-fitting term. It can be seen that TVX^0 minimizes the number of corners of the shape in U and hence leads to a gradually simplified polygonal approximation of the original shape. TVX^1 minimizes the total curvature of the shape in U and hence leads to a piecewise smooth approximation of the shape.

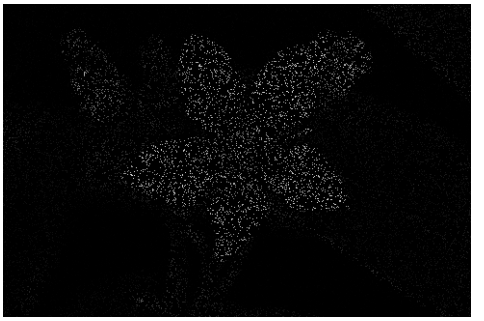
In Figure 7.15 we provide a visualization of the measure μ in the roto-translation space for the image shown in Figure 7.14(e), obtained using TVX^0 regularization. One can observe that in our discrete approximation the measure μ nicely concentrates on thin lines in the roto-translation space.

In our second experiment we consider image inpainting. For this we choose

$$g(U) = \sum_{(i,j)\in\mathcal{I}} \delta_{\{U^\diamond_{i,j}\}}(U_{i,j}),$$

where $U^\diamond \in \mathbb{R}^{m\times n}$ is a given image and $\mathcal{I}$ defines the set of indices for which pixel information is available. Figure 7.16 shows the image inpainting results, where we have used the same test image as in Figure 7.7. The parameters α, β of the TVX model were set to $\alpha = 0.01$, $\beta = 1$, and we

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) 90% missing data

(b) 80% missing lines

(c) TVX^1 inpainting

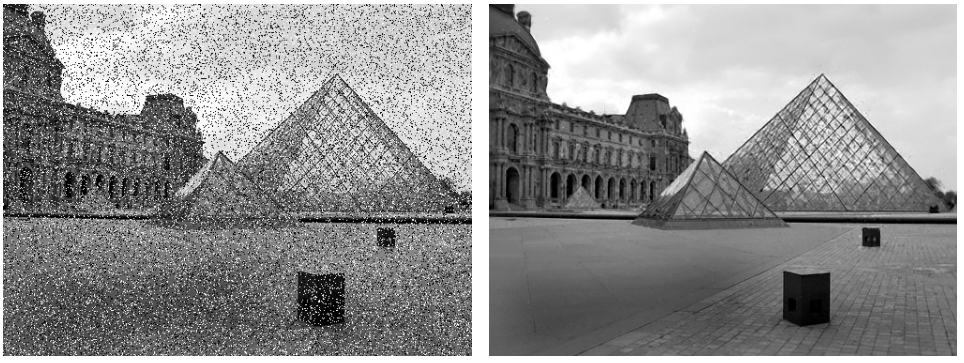
(d) TVX^1 inpainting

Figure 7.16. Image inpainting using TVX^1 regularization. (a,c,e) Input image with 90% missing pixels and recovered solutions. (b,d,f) Input image with 80% missing lines and recovered solutions.

used $o = 32$ discrete orientations. The parameter α is used to control the amount of total variation regularization while the parameter β is used to control the amount of curvature regularization. We tested two different kinds of missing pixel information. In the experiment shown on the left we randomly threw away 90% of the image pixels, whereas in the experiment shown on the right we skipped 80% of entire rows of the image. From the results one can see that the TVX^1 models can faithfully reconstruct the missing image information even if there are large gaps.

In our third experiment we apply the TVX^1 regularizer for image denoising in the presence of salt-and-pepper noise. Following the classical TV-ℓ_1 model, we used a data term based on the ℓ_1-norm: $g(U) = \lambda\|U - U^\diamond\|_1$, where $U^\diamond$ is the given noisy image. We applied the TVX^1 model to the same test image as used in Figure 2.3. In this experiment the parameters for the regularizer were set to $\alpha = 0.01$, $\beta = 1$. For the data term we used $\lambda = 0.25$, and we used $o = 32$ discrete orientations. Figure 7.17 shows the results obtained by minimizing the TVX^1 model. The result shows that the

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) noisy image (b) TVX^1 denoised

Figure 7.17. Denoising an image containing salt-and-pepper noise. (a) Noisy image degraded by 20% salt-and-pepper noise. (b) Denoised image using TVX^1 regularization. Note the significant improvement over the result of the TV-ℓ_1 model, shown in Figure 2.3.

TVX^1 model performs particularly well at preserving thin and elongated structures (*e.g.* on the glass pyramid). The main reason why these models works so well when applied to salt-and-pepper denoising is that the problem is actually very close to inpainting, for which curvature minimizing models were originally developed.

7.12. Lasso and dictionary learning

Let us return to one of the first sparse models considered in this paper, the Lasso model (2.2). We will describe how the Lasso model can be used for image denoising. This technique is related to state-of-the art non-local approaches for image denoising, although somewhat simplified (see the references in the Introduction). Let $p = (p_1, \dots, p_k)$ be a set of k image patches, where each patch p_i, $i = 1 \dots, k$ is of size $m \times n$ pixels. From the patches we form a matrix $P = (P_1, \dots, P_k) \in \mathbb{R}^{mn \times k}$, where $P_i \in \mathbb{R}^{mn}$ is a vectorized version of the patch p_i. Moreover, we consider a dictionary $D = (D_1, \dots, D_l) \in \mathbb{R}^{mn \times l}$, where each $D_j \in \mathbb{R}^{mn}$, $j = 1, \dots, l$ are the atoms of the dictionary. Our goal is now to find a dictionary that allows for a sparse representation of the entire set of patches. The Lasso model specialized to this setting is given by

$$\min_{X,D} \lambda \|X\|_1 + \frac{1}{2}\|DX - P\|_2^2,$$

where $X = (X_1, \dots, X_l) \in \mathbb{R}^{l \times k}$ is a matrix holding the coefficients of the dictionary and $\lambda > 0$ is the regularization parameter. The basic idea behind learning an optimal dictionary is based on minimizing the Lasso problem

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

with respect to both X and D. See for example Olshausen and Field (1997) for one of the first attempts, Aharon *et al.* (2006) for an algorithm based on the singular value decomposition, Lee, Battle, Raina and Ng (2007) for an algorithm based on alternating minimization, and Mairal, Bach, Ponce and Sapiro (2009*a*) for a stochastic (online) algorithm. It is now acknowledged that very large-scale Lasso problems should rather be tackled by accelerated stochastic descent algorithms, for instance the accelerated proximal coordinate gradient method (APCG) of Lin *et al.* (2015).

Here we directly minimize the Lasso objective using the PALM algorithm (Algorithm 12). In order to eliminate scaling ambiguities, however, it is necessary to put constraints on the atoms of the dictionary. We found it useful to require that all but the first atom D_1 should have zero mean and a 2-norm less than or equal to one. The reason why we do not put any constraints on the first atom is that we expect the first atom to capture the low-frequency part of the patches and the other atoms should capture the high frequencies. Hence we consider the following convex set:

$$C = \{D \in \mathbb{R}^{mn \times l} : 1^T D_j = 0,\ \|D_j\|_2 \leq 1,\ j = 2, \dots, l\}.$$

Further, since we do not expect the patches to be sparse in the low-frequency atom, we penalize the ℓ_1-norm only for the coefficients corresponding to the high-frequency patches. Using these constraints, the final dictionary learning problem takes the form

$$\min_{X,D} \lambda \sum_{j=2}^{l} \|X_j\|_1 + \frac{1}{2}\|DX - P\|_2^2, \quad \text{such that } D \in C. \tag{7.37}$$

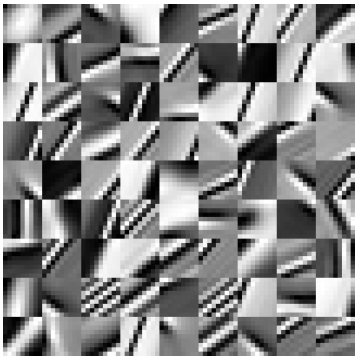
This problem is non-convex but it has a relatively nice structure, which makes Algorithm 12 suitable. Indeed the data-fitting term is smooth (quadratic) and has partially Lipschitz-continuous gradients. Furthermore, the proximal maps with respect to both the ℓ_1-norm and the convex set C are easy to implement. The proximal map with respect to the ℓ_1-norm can be computed by the usual soft-shrinkage formula. The projection of the dictionary onto C is computed independently for each patch by first removing the average of each D_j, and then projecting onto the 2-ball with radius 1. In our experiments we used an inertial variant of the PALM algorithm, whose convergence has recently been analysed in Pock and Sabach (2016).

Figure 7.18 illustrates an experiment in which we learned an optimal dictionary on the clean image and then applied the learned dictionary to denoise a noisy version of the image. During learning, we set $\lambda = 0.1$, we used a patch size of $m = n = 9$, and the number of dictionary atoms was set to $l = 81$. The size of the image is 327×436, from which we extracted roughly $k = 136\,500$ patches using a sliding-window approach. One can see that the learned dictionary nicely captures the most frequent structures

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) original image g

(b) noisy image f



(c) dictionary

(d) denoised image (PSNR = 26.57)

Figure 7.18. Image denoising using a patch-based Lasso model. (a) Original image, and (b) its noisy variant, where additive Gaussian noise with standard deviation 0.1 has been added. (c) Learned dictionary containing 81 atoms with patch size 9×9, and (d) final denoised image.

in the image. The dictionary atom shown top left corresponds to the low-frequency part of the patches. It can be seen from the figure that this atom does not have a zero mean.

After learning was completed, we applied the learned dictionary to the restoration of a noisy version of the original image.[22] We first extracted the patches $\tilde{p}_i \in \mathbb{R}^{m\times n}$, $i = 1, \dots, k$ from the noisy image and again arranged the patches into a matrix $\tilde{P} \in \mathbb{R}^{mn\times k}$. In order to denoise the set of patches,

[22] A more reasonable approach would of course be to learn the dictionary on a set of representative images (excluding the test image). Although we learn the dictionary on the patches of the original image, observe that we are still far from obtaining a perfect reconstruction. On one hand the number of dictionary atoms (81) is relatively small compared to the number of patches (136 500), and on the other hand the regularization parameter also prevents overfitting.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

we solved the Lasso problem (7.37), but this time only with respect to the coefficient vector X. This problem is convex and can be solved using Algorithm 5 (FISTA). The parameter λ was set to $\lambda = 0.1$. After the patches were denoised, we reconstructed the final image by averaging over all the patches. The reconstructed image is shown in Figure 7.18(d). Observe that the texture and elongated structures are nicely preserved in the denoised image.

Another interesting variant of the patch-based Lasso problem – particularly in the context of image processing – is given by a convolutional sparse model (Zeiler, Krishnan, Taylor and Fergus 2010) of the form

$$\min_{(v_i)_{i=1}^k} \sum_{i=1}^{k} \|v_i\|_1 + \frac{\lambda}{2} \left\| \sum_{i=1}^{k} d_i * v_i - u^\diamond \right\|_2^2, \tag{7.38}$$

where $u^\diamond \in \mathbb{R}^{m\times n}$ is an input image, $v_i \in \mathbb{R}^{m\times n}$, $i = 1, \dots, k$ are a set of k sparse coefficient images, and $d_i \in \mathbb{R}^{l\times l}$, $i = 1, \dots, k$ are the corresponding two-dimensional filter kernels. The approximated image $u \in \mathbb{R}^{m\times n}$ can be recovered via $u = \sum_{i=1}^{k} d_i * v_i$. The main advantage of the convolutional model over the patch-based model is that the convolution operation inherently models the translational invariance of images. This model is also strongly related to the layers adopted in recently proposed convolutional neural networks (CNNs), which have been shown to perform extremely well on large-scale image classification tasks (Krizhevsky *et al.* 2012).

For learning the filters d_i, we minimize the convolutional Lasso problem (7.38) with respect to both the filters d_i and the coefficient images v_i. Some care has to be taken to avoid a trivial solution. Therefore we fix the first filter kernel to be a Gaussian filter and fix the corresponding coefficient image to be the input image $u^\diamond$. Hence, the problem is equivalent to learning the dictionary only for the high-frequency filtered image $\tilde{u} = u^\diamond - g * u^\diamond$, where $g \in \mathbb{R}^{l\times l}$ is a Gaussian filter with standard deviation $\sigma = l$.

To minimize (7.38) in v_i and d_i, we again use the inertial variant of the PALM algorithm. We used $k = 81$ filters of size $l = 9$ and the first filter was set to a Gaussian filter of the same size. The regularization parameter λ was set to $\lambda = 0.2$. Figure 7.19(a) shows the filters we have learned on the clean image shown in Figure 7.18(a). Comparing the learned convolution filters to the dictionary of the patch-based Lasso problem, one can see that the learned filters contain Gabor-like structures (Hubel and Wiesel 1959) but also more complex structures, which is a known effect caused by the induced shift invariance (Hashimoto and Kurata 2000). We then also applied the convolutional Lasso model to a noisy variant of the original image, and the result is shown in Figure 7.19(b). From the PSNR values, one can see that the convolutional Lasso model leads to a slightly better result.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) learned filters

(b) denoised image (PSNR = 26.68)

Figure 7.19. Image denoising using the convolutional Lasso model. (a) The 81 convolution filters of size 9×9 that have been learned on the original image. (b) Denoised image obtained by minimizing the convolutional Lasso model.

7.13. Support vector machine

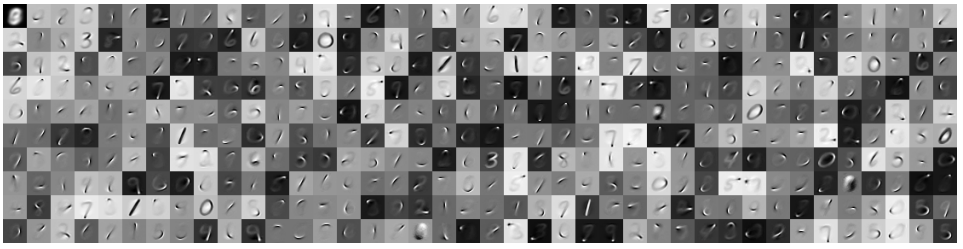
Linear classification in a feature space is an important problem in many scientific disciplines. See for example the review paper by Burges (1998) and references therein. The general idea is as follows. Let $\{x_1, \dots, x_n\} \subset \mathbb{R}^d$ be a set of d-dimensional feature vectors with corresponding positive and negative class labels $y_i \in \{-1, +1\}$. The idea of a linear classifier is to find a separating hyperplane $w \in \mathbb{R}^d$ and a bias term $b \in \mathbb{R}$, such that the feature vectors corresponding to class $+1$ are (in some sense) most robustly separated by the feature vectors corresponding to class labels -1. This problem can be written as the minimization of the following loss function, known as the *support vector machine* (SVM) (Vapnik 2000):

$$\min_{w,b} \frac{1}{2}\|w\|^2 + \frac{C}{n}\sum_{i=1}^{n} h(y_i(\langle w, x_i\rangle + b)), \tag{7.39}$$

where $\frac{1}{2}\|w\|^2$ is a regularization term that avoids overfitting to the training data, $h(\cdot)$ is the Hinge-loss function defined by $h(t) = \max\{0, 1 - t\}$, and $C > 0$ is a regularization parameter. Note that the Hinge-loss function is closely related to the ℓ_1-norm and hence it also induces sparsity in the solution – the idea being that it will minimize the number of samples which are within a certain margin around the hyperplane ($t \in [0, 1]$) or are wrongly classified ($t < 0$). If the feature space is linearly separable, the SVM returns the hyperplane that maximizes the distance to the closest feature example (which is half the 'margin'), which can be interpreted as minimizing the risk of misclassifying any new feature example. To simplify the problem, we can replace the bias term with an additional constant $(d + 1)$th component

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) subset of the MNIST database



(b) dictionary containing 400 atoms

Figure 7.20. MNIST training images and dictionary.

added to each sample x_i, which corresponds to assuming that all samples 'live' in a hyperplane far from the origin, and therefore has a very similar effect (the bias b is replaced with cy_iw_{d+1} for some constant c of the order of the norm of the samples). This smoothing makes the problem strongly convex, hence slightly easier to solve, as one can use Algorithm 8. An additional acceleration trick consists in starting the optimization with a small number of samples and periodically adding to the problem a fraction of the worst classified samples. As it is well known (and desirable) that only a small proportion of the samples should be really useful for classification (the 'support vectors' which bound the margin), it is expected, and actually observed, that the size of the problems can remain quite small with this strategy.

An extension of the SVM to non-linear classifiers can be achieved by applying the kernel trick (Aĭzerman, Braverman and Rozonoèr 1964) to the hyperplane, which lifts the linear classifier to a new feature space of arbitrary (even infinite) dimension (Vapnik 2000).

To illustrate this method, we have tried to learn a classifier on the 60 000 digits of the MNIST[23] database (LeCun, Bottou, Bengio and Haffner 1998*a*): see Figure 7.20. Whereas it is known that a kernel SVM can achieve good performance on this dataset (see the results reported on the web page of the project) it is computationally quite expensive, and we have tried here to

[23] http://yann.lecun.com/exdb/mnist

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

incorporate non-linearities in a simpler way. To start with, it is well known that training a linear SVM directly on the MNIST data (which consists of small 28×28 images) does not lead to good results. To improve the performance, we trained the 400-component dictionary shown in Figure 7.20, using the model in Section 7.12, and then computed the coefficients $((c_i)_{i=1}^{400})$ of each MNIST digit on this dictionary using the Lasso problem. This represents a fairly large computation and may take several hours on a standard computer.

Then we trained the SVMs on feature vectors of the form, for each digit, $(\tilde{c}, (c_i)_{i=1}^{400}, (c_i^2)_{i=1}^{400})$ (in dimension 801), where $\tilde{c}$ is the constant which maps all vectors in a hyperplane 'far' from the origin, as explained above, and the additional $(c_i^2)_{i=1}^{400}$ represent a non-linear lifting which slightly boosts the separability of the vectors. This mimics a non-linear kernel SVM with a simple isotropic polynomial kernel.

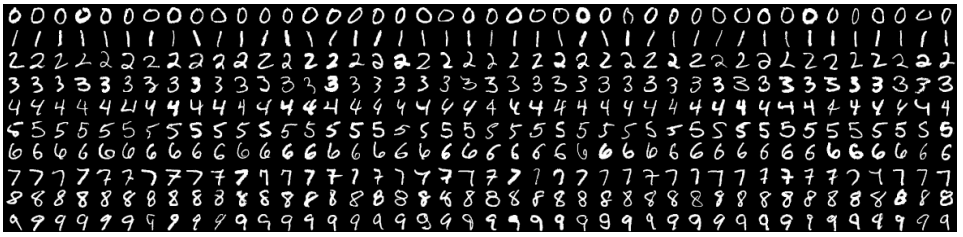
The technique we have employed here is a standard 'one-versus-one' classification approach, which proved slightly more efficient than, for instance, training an SVM to separate each digit from the rest. It consists in training 45 vectors $w_{i,j}$, $0 \le i < j \le 9$, each separating the training subset of digits i from the digits j (in this case, in particular, each learning problem remains quite small).

Then, to classify a new digit, we have counted how often it is classified as 'i' or 'j' by $w_{i,j}$ (which is simply testing whether $\langle w_{i,j}, x_i \rangle$ is positive or negative). If everything were perfect, a digit would be classified nine times as its true label, hence it is natural to consider that the label that gets the maximal number of 'votes' is the expected one.[24] This very elementary approach leads to an error of 2.21% on the 10 000 test digits of the database, which is much worse than the best results reached so far but quite reasonable for such a simple approach. Let us observe that for each failed classification, the second vote was correct except for 70 digits, that is, 0.7% of the base: hence it is not surprising that more elaborate representations or classification techniques can reach classification error rates of this order: 0.49% for SVM methods based on 'scattering' networks (Bruna and Mallat 2013) and 0.23% for the most recent CNN-based methods (Ciresan, Meier and Schmidhuber 2012). The failed classifications are shown in Figure 7.21, as well as some of the well-recognized digits.

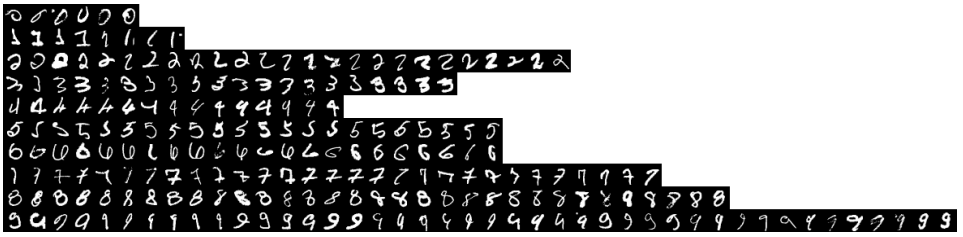
An interesting extension to the unsupervised dictionary learning problem consists in augmenting the dictionary learning objective function with a loss function, ensuring that the learned features not only lead to a sparse representation but can also be well classified with the SVM; see Mairal *et al.* (2009*b*). This supervised dictionary learning actually improves the final classification results (leading to an error of only 0.6%).

[24] This can be slightly improved: see for instance Duan and Keerthi (2005).

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press



(a) subset of the well-classified digits



(b) the 221 (out of 10 000) wrongly classified digits

Figure 7.21. MNIST classification results.

7.14. Non-linear deconvolution: inverting a CNN

In this last example, we show an interesting application of non-linear deconvolution to invert a pre-trained convolutional neural network. The CNN we consider here is a state-of-the-art very deep CNN that has been trained on the ImageNet (Krizhevsky *et al.* 2012) classification problem (Simonyan and Zisserman 2015). Whereas it has been shown by Bruna, Szlam and LeCun (2014) that some CNN structures are invertible, it is not clear whether this particular one is. Let $C : \mathbb{R}^{m\times n\times 3} \to \mathbb{R}^k$ denote a non-linear map that takes a colour image $\mathbf{u}$ of size $m \times n$ pixels as input and produces a feature vector ϕ of length k. The task is now to start from a given feature vector $\phi^\diamond$ and try to find an image $\mathbf{u}$ such that $C(\mathbf{u}) \approx \phi^\diamond$. Since this problem is ill-posed, we aim at minimizing the following total variation regularized problem (Mahendran and Vedaldi 2015):

$$\min_{\mathbf{u}} \lambda\|\mathbf{D}\mathbf{u}\|_{2,1} + \frac{1}{2}\|C(\mathbf{u}) - \phi^\diamond\|^2.$$

We minimize the energy by performing a gradient descent using the iPiano algorithm (Algorithm 11). For this we replace the total variation regularizer with its smooth Huber variant (4.18). The gradient with respect to the data term can be computed by using back-propagation.

Figure 7.22 shows the results we get by first computing a 4096-dimensional feature vector $\phi^\diamond = C(\mathbf{u}^\diamond)$ from the original image $\mathbf{u}^\diamond$ and then inverting this feature vector by solving the non-linear deconvolution problem. We

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

(a) input image (b) recovered image

Figure 7.22. Inverting a convolutional neural network. (a) Original image used to compute the initial feature vector $\phi^\diamond$. (b) Image recovered from the non-linear deconvolution problem. Due to the high degree of invariances of the CNN with respect to scale and spatial position, the recovered image contains structures from the same object class, but the image looks very different.

initialized the algorithm with an image $\mathbf{u}$ obtained by filtering the original image with Gaussian noise. One can see that the structures that are 'hallucinated' by the CNN are of the same class of objects as the original image, but the algorithm produces a completely new, yet almost reasonable image.

Acknowledgements

The authors benefit from support of the ANR and FWF via the 'EANOI' (Efficient Algorithms for Nonsmooth Optimization in Imaging) joint project, FWF no. I1148 / ANR-12-IS01-0003. Thomas Pock also acknowledges the support of the Austrian Science Fund (FWF) under the START project BIVISION, no. Y729, and the European Research Council under the Horizon 2020 program, ERC starting grant 'HOMOVIS', no. 640156. Antonin Chambolle also benefits from support of the 'Programme Gaspard Monge pour l'Optimisation et la Recherche Opérationnelle' (PGMO), through the 'MAORI' group, as well as the 'GdR MIA' of the CNRS. He also warmly thanks Churchill College and DAMTP, Centre for Mathematical Sciences, University of Cambridge, for their hospitality, with the support of the French Embassy in the UK. Finally, the authors are very grateful to Yunjin Chen, Jalal Fadili, Yura Malitsky, Peter Ochs and Glennis Starling for their comments and their careful reading of the manuscript.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

A. Abstract convergence theory

We recall an essential result on weak contractions (or non-expansive mappings) in Hilbert spaces. More details can be found in Bauschke and Combettes (2011) or the recent review by Burger *et al.* (2014). As mentioned in Section 4.1, for convex f the operator

$$x \mapsto x - \tau \nabla f(x)$$

is an 'averaged operator' when $\tau \in (0, 2/L)$, where L is the Lipschitz constant of ∇f: this means that it is of the form

$$T_\theta x := \theta x + (1-\theta) T_0 x$$

with $\theta \in (0,1)$, where T_0 is a weak contraction, satisfying $\|T_0 x - T_0 y\| \leq \|x - y\|$ for all x, y. Indeed, let $\theta = \tau L/2$ and $T_0 x := x - (2/L)\nabla f(x)$ to recover this fact. Let F denote the set of fixed points of T_0, that is, $F = \{x \in \mathcal{X} : T_0 x = x\}$. Then we obtain the following result, usually called the Krasnosel'skii–Mann theorem: see Bauschke and Combettes (2011, Theorem 5.13) and Bertsekas (2015, Theorem 5.1.9).

Theorem A.1. Let $x \in \mathcal{X}$, $0 < \theta < 1$, and assume $F \neq \emptyset$. Then $(T_\theta^k x)_{k \geq 1}$ weakly converges to some point $x^* \in F$.

Proof. Throughout this proof let $x_k = T_\theta^k x$ for each $k \geq 0$.

Step 1. The first observation is that since T_θ is also a weak contraction, the sequence $(\|x_k - x^*\|)_k$ is non-increasing for any $x^* \in F$ (which is also the set of fixed points of T_θ). The sequence $(x_k)_k$ is said to be *Fejér-monotone* with respect to F, which yields a lot of interesting consequences; see Bauschke and Combettes (2011, Chapter 5) for details. It follows that for any $x^* \in F$, one can define $m(x^*) := \inf_k \|x_k - x^*\| = \lim_k \|x_k - x^*\|$. If there exists x^* such that $m(x^*) = 0$, then the theorem is proved, as x_k converges strongly to x^*.

Step 2. If not, let us show that we still obtain $T_\theta x_k - x_k = x_{k+1} - x_k \to 0$. An operator which satisfies this property is said to be *asymptotically regular* (Browder and Petryshyn 1966). We will use the following result, which is standard, and in fact gives a hint that this proof can be extended to more general spaces with uniformly convex norms.

Lemma A.2. For all $\varepsilon > 0$, $\theta \in (0,1)$, there exists $\delta > 0$ such that, for all $x, y \in \mathcal{X}$ with $\|x\|, \|y\| \leq 1$ and $\|x - y\| \geq \varepsilon$,

$$\|\theta x + (1-\theta) y\| \leq (1 - \delta) \max\{\|x\|, \|y\|\}.$$

This follows from the strong convexity of $x \mapsto \|x\|^2$ (*i.e.* the parallelogram identity), and we leave the proof to the reader.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Now assume that along a subsequence, we have $\|x_{k_l+1} - x_{k_l}\| \geq \varepsilon > 0$. Observe that

$$x_{k_l+1} - x^* = \theta(x_{k_l} - x^*) + (1-\theta)(T_0 x_{k_l} - x^*)$$

and that

$$(x_{k_l} - x^*) - (T_0 x_{k_l} - x^*) = x_{k_l} - T_0 x_{k_l} = -\frac{1}{1-\theta}(x_{k_l+1} - x_{k_l}),$$

so that

$$\|(x_{k_l} - x^*) - (T_0 x_{k_l} - x^*)\| \geq \varepsilon/(1-\theta) > 0.$$

Hence we can invoke the lemma (remember that $(x_k - x^*)_k$ is globally bounded since its norm is non-increasing), and we obtain that, for some $\delta > 0$,

$$m(x^*) \leq \|x_{k_l+1} - x^*\| \leq (1-\delta)\max\{\|x_{k_l} - x^*\|, \|T_0 x_{k_l} - x^*\|\},$$

but since $\|T_0 x_{k_l} - x^*\| \leq \|x_{k_l} - x^*\|$, it follows that

$$m(x^*) \leq (1-\delta)\|x_{k_l} - x^*\|.$$

As $k_l \to \infty$, we get a contradiction if $m(x^*) > 0$.

Step 3. Assume now that $\bar{x}$ is the weak limit of some subsequence $(x_{k_l})_l$. Then we claim it is a fixed point. An easy way to see it is to use Minty's trick (Brézis 1973) and the fact that $I - T_\theta$ is a monotone operator. Another is to use Opial's lemma.

Lemma A.3 (Opial 1967, Lemma 1). If the sequence $(x_n)_n$ is weakly convergent to x_0 in a Hilbert space $\mathcal{X}$, then, for any $x \neq x_0$,

$$\liminf_n \|x_n - x\| > \liminf_n \|x_n - x_0\|.$$

The proof in the Hilbert space setting is easy and we leave it to the reader. Since T_θ is a weak contraction, we observe that for each k,

$$\begin{aligned}\|x_k - \bar{x}\|^2 &\geq \|T_\theta x_k - T_\theta \bar{x}\|^2 \\ &= \|x_{k+1} - x_k\|^2 + 2\langle x_{k+1} - x_k, x_k - T_\theta \bar{x}\rangle + \|x_k - T_\theta \bar{x}\|^2,\end{aligned}$$

and we deduce (thanks to Step 2 above)

$$\liminf_l \|x_{k_l} - \bar{x}\| \geq \liminf_l \|x_{k_l} - T_\theta \bar{x}\|.$$

Opial's lemma implies that $T_\theta \bar{x} = \bar{x}$. One advantage of this approach is that it can be easily extended to a class of Banach spaces (Opial 1967).

Step 4. To conclude, assume that a subsequence $(x_{m_l})_l$ of $(x_n)_n$ converges weakly to another fixed point $\bar{y}$. Then it must be that $\bar{y} = \bar{x}$, otherwise Opial's Lemma A.3 would again imply that $m(\bar{x}) < m(\bar{y})$ and $m(\bar{y}) < m(\bar{x})$. It follows that the whole sequence (x_n) must converge weakly to $\bar{x}$. □

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

The notion of averaged operators dates back at least to Schaefer (1957), Krasnosel'skiĭ (1955) (with $\theta = 1/2$) and Mann (1953) (with possibly arbitrarily long averages). Forms of this classical result were proved in the first two papers (usually in a more general context such as Banach spaces) and many others (Opial 1967), as well as variants, such as with varying operators[25] (Browder 1967). This and many useful extensions can be found in Bauschke and Combettes (2011). The special case $\theta \geq 1/2$ is the case of 'firmly non-expansive operators', which is well known to coincide with the proximity operators and the resolvent $(I + A)^{-1}$ of maximal-monotone operators (see the review in Bauschke *et al.* 2012). Another important observation is that the composition of two averaged operators is, again, an averaged operator; this is straightforward and we leave the proof to the reader. It implies the convergence of forward–backward splitting and in fact of many similar splitting techniques; see Combettes (2004) for further results in this direction. Finally, we mention that one can improve such results to obtain convergence rates; in particular, Liang *et al.* (2015) have recently shown that for some problems one can get an eventual linear convergence for algorithms based on this type of iteration.

B. Proof of Theorems 4.1, 4.9 and 4.10.

Here we prove the rates of convergence for a class of accelerated descent algorithms introduced in Nesterov (2004) and Beck and Teboulle (2009). We give proofs which differ slightly from the classical ones, and unify both presentations. For the FISTA algorithm the proof presented here is found in a few places such as the references Chambolle and Pock (2015*b*), Chambolle and Dossal (2015), Burger *et al.* (2014), Bertsekas (2015) and Bonettini *et al.* (2015). As Theorem 4.1 is a particular case of Theorem 4.9 (with $g = 0$), we just prove the latter.

Proof of Theorem 4.9. We start from inequality (4.37), letting, $\bar{x} = x^k$ and $\hat{x} = x^{k+1}$ for $k \geq 0$. It follows that, for any x,

$$F(x) + (1 - \tau\mu_f)\frac{\|x - x^k\|^2}{2\tau} \geq F(x^{k+1}) + (1 + \tau\mu_g)\frac{\|x - x^{k+1}\|^2}{2\tau}.$$

Choosing $x = x^k$ shows that $F(x^k)$ is non-increasing. Letting

$$\omega = \frac{1 - \tau\mu_f}{1 + \tau\mu_g} \leq 1$$

[25] The proof above can easily be extended to allow for some variation of the averaging parameter θ. This would yield convergence, for instance, for gradient descent algorithms with varying steps (within some bounds) and many other similar methods.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

and summing these inequalities from $k=0$ to $n-1$, $n \geq 1$, after multiplication by ω^{-k-1}, we find

$$\sum_{k=1}^{n} \omega^{-k}(F(x^k) - F(x)) + \sum_{k=1}^{n} \omega^{-k} \frac{1+\tau\mu_g}{2\tau}\|x - x^k\|^2 \leq \sum_{k=0}^{n-1} \omega^{-k-1} \frac{1-\tau\mu_f}{2\tau}\|x - x^k\|^2.$$

After cancellations, and using $F(x^k) \geq F(x^n)$ for $k = 0, \dots, n$, we get

$$\omega^{-n}\left(\sum_{k=0}^{n-1} \omega^k\right)(F(x^n) - F(x)) + \omega^{-n} \frac{1+\tau\mu_g}{2\tau}\|x - x^n\|^2 \leq \frac{1+\tau\mu_g}{2\tau}\|x - x^0\|^2.$$

We deduce both (4.29) (for $\mu = \mu_f + \mu_g > 0$ so that $\omega < 1$) and (4.28) (for $\mu = 0$ and $\omega = 1$). □

Proof of Theorem 4.10. The idea behind the proof of Beck and Teboulle (2009) is to improve this inequality (4.37) by trying to obtain strict decay of the term in F in the inequality. The trick is to use (4.37) at a point which is a convex combination of the previous iterate and an arbitrary point.

If, in (4.37), we replace x with $((t-1)x^k + x)/t$ ($t \geq 1$), $\bar{x}$ with y^k and $\hat{x}$ with $x^{k+1} = T_\tau y^k$, where $t \geq 1$ is arbitrary, we find that for any x (after multiplication by t^2),

$$\begin{aligned} t(t-1)(F(x^k) - F(x)) - \mu\frac{t-1}{2}\|x - x^k\|^2 \\ + (1-\tau\mu_f)\frac{\|(t-1)x^k + x - ty^k\|^2}{2\tau} \\ \geq t^2(F(x^{k+1}) - F(x)) + (1+\tau\mu_g)\frac{\|(t-1)x^k + x - tx^{k+1}\|^2}{2\tau}. \end{aligned} \tag{B.1}$$

Then we observe that

$$\begin{aligned} &-\mu\frac{t-1}{2}\|x - x^k\|^2 + (1-\tau\mu_f)\frac{\|x - x^k + t(x^k - y^k)\|^2}{2\tau} \\ &\quad = (1 - \tau\mu_f - \mu\tau(t-1))\frac{\|x - x^k\|^2}{2\tau} + \frac{1-\tau\mu_f}{\tau}t\langle x - x^k, x^k - y^k\rangle \\ &\qquad + t^2(1-\tau\mu_f)\frac{\|x^k - y^k\|^2}{2\tau} \\ &\quad = \frac{(1+\tau\mu_g - t\mu\tau)}{2\tau}\left\|x - x^k + t\frac{1-\tau\mu_f}{1+\tau\mu_g - t\mu\tau}(x^k - y^k)\right\|^2 \\ &\qquad + t^2(1-\tau\mu_f)\left(1 - \frac{1-\tau\mu_f}{1+\tau\mu_g - t\mu\tau}\right)\frac{\|x^k - y^k\|^2}{2\tau} \end{aligned}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

$$= \frac{(1+\tau\mu_g - t\mu\tau)}{2\tau}\left\|x - x^k + t\frac{1-\tau\mu_f}{1+\tau\mu_g-t\mu\tau}(x^k - y^k)\right\|^2 - t^2(t-1)\frac{\tau\mu(1-\tau\mu_f)}{1+\tau\mu_g-t\mu\tau}\frac{\|x^k-y^k\|^2}{2\tau}.$$

It follows that, for any $x \in \mathcal{X}$,

$$\begin{aligned} t(t-1)(F(x^k)-F(x)) + (1+\tau\mu_g-t\mu\tau)\frac{\|x - x^k - t\frac{1-\tau\mu_f}{1+\tau\mu_g-t\mu\tau}(y^k-x^k)\|^2}{2\tau} \\ \geq t^2(F(x^{k+1})-F(x)) + (1+\tau\mu_g)\frac{\|x-x^{k+1}-(t-1)(x^{k+1}-x^k)\|^2}{2\tau} \\ + t^2(t-1)\frac{\tau\mu(1-\tau\mu_f)}{1+\tau\mu_g-t\mu\tau}\frac{\|x^k-y^k\|^2}{2\tau}. \end{aligned} \tag{B.2}$$

We let $t = t_{k+1}$ above. Then we can get a useful recursion if we let

$$\omega_k = \frac{1+\tau\mu_g - t_{k+1}\mu\tau}{1+\tau\mu_g} = 1 - t_{k+1}\frac{\mu\tau}{1+\tau\mu_g} \in [0,1], \tag{B.3}$$

$$t_{k+1}(t_{k+1}-1) \leq \omega_k t_k^2, \tag{B.4}$$

$$\beta_k = \frac{t_k - 1}{t_{k+1}}\frac{1+\tau\mu_g - t_{k+1}\mu\tau}{1-\tau\mu_f} = \omega_k\frac{t_k-1}{t_{k+1}}\frac{1+\tau\mu_g}{1-\tau\mu_f}, \tag{B.5}$$

$$y^k = x^k + \beta_k(x^k - x^{k-1}). \tag{B.6}$$

Denoting $\alpha_k = 1/t_k$ and

$$q = \frac{\tau\mu}{1+\tau\mu_g} = \frac{\tau\mu_f + \tau\mu_g}{1+\tau\mu_g} < 1,$$

we easily check that these rules are precisely the same as in Nesterov (2004, formula (2.2.9), p. 80), with the minor difference that in our case the choice $t_0 = 0, t_1 = 1$ is admissible[26] and a shift in the numbering of the sequences (x^k), (y^k). In this case we find

$$\begin{aligned} t_{k+1}^2(F(x^{k+1})-F(x)) + \frac{1+\tau\mu_g}{2\tau}\|x - x^{k+1} - (t_{k+1}-1)(x^{k+1}-x^k)\|^2 \\ \leq \omega_k\left(t_k^2(F(x^k)-F(x)) + \frac{1+\tau\mu_g}{2\tau}\|x - x^k - (t_k-1)(x^k - x^{k-1})\|^2\right), \end{aligned}$$

[26] Note, however, that this is no different from performing a first step of the forward–backward descent scheme to the energy before actually implementing Nesterov's iterations.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

so that

$$t_k^2(F(x^k) - F(x)) + \frac{1+\tau\mu_g}{2\tau}\|x - x^k - (t_k - 1)(x^k - x^{k-1})\|^2$$
$$\leq \left(\prod_{n=0}^{k-1} \omega_n\right)\left[t_0^2(F(x^0) - F(x)) + \frac{1+\tau\mu_g}{2\tau}\|x - x^0\|^2\right]. \tag{B.7}$$

The update rule for t_k reads

$$t_{k+1}(t_{k+1} - 1) = (1 - qt_{k+1})t_k^2, \tag{B.8}$$

so that

$$t_{k+1} = \frac{1 - qt_k^2 + \sqrt{(1 - qt_k^2)^2 + 4t_k^2}}{2}. \tag{B.9}$$

We need to make sure that $qt_{k+1} \leq 1$, so that (B.3) holds. This is proved exactly as in the proof of Lemma 2.2.4 of Nesterov (2004). Assuming (as in Nesterov 2004) that $\sqrt{q}t_k \leq 1$, we observe that (B.8) yields

$$qt_{k+1}^2 = qt_{k+1} + (1 - qt_{k+1})qt_k^2.$$

If $qt_{k+1} \geq 1$, then $qt_{k+1}^2 \leq qt_{k+1}$, and hence $qt_{k+1} \leq q < 1$, a contradiction. Hence $qt_{k+1} < 1$ and we obtain that qt_{k+1}^2 is a convex combination of 1 and qt_k^2, so that $\sqrt{q}t_{k+1} \leq 1$. We have shown that when $\sqrt{q}t_0 \leq 1$, which we will now assume, $\sqrt{q}t_k \leq 1$ for all k. Finally, we also observe that

$$t_{k+1}^2 = (1 - qt_k^2)t_{k+1} + t_k^2,$$

showing that t_k is an increasing sequence. It remains to estimate the factor

$$\theta_k = t_k^{-2}\prod_{n=0}^{k-1} \omega_n \quad \text{for } k \geq 1.$$

From (B.4) (with an equality) we find that

$$1 - \frac{1}{t_{k+1}} = \omega_k \frac{t_k^2}{t_{k+1}^2},$$

so

$$t_0^2\theta_k = \frac{t_0^2}{t_k^2}\prod_{n=0}^{k-1} \omega_n = \prod_{n=1}^{k}\left(1 - \frac{1}{t_k}\right) \leq (1 - \sqrt{q})^k$$

since $1/t_k \geq \sqrt{q}$. If $t_0 \geq 1$, then $\theta_k \leq (1 - \sqrt{q})^k/t_0^2$. If $t_0 \in [0, 1[$, we instead write

$$\theta_k = \frac{\omega_0}{t_k^2}\prod_{n=1}^{k-1} \omega_n = \frac{\omega_0}{t_1^2}\prod_{n=2}^{k}\left(1 - \frac{1}{t_k}\right)$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

and observe that (B.9) yields (using $2 - q \geq 1 \geq q$)

$$t_1 = \frac{1 - qt_0^2 + \sqrt{1 + 2(2-q)t_0^2 + q^2 t_0^4}}{2} \geq 1.$$

Also, $\omega_0 \leq 1 - q$ (from (B.3)), so that

$$\theta_k \leq (1 + \sqrt{q})(1 - \sqrt{q})^k.$$

The next step is to bound θ_k by $O(1/k^2)$. It also follows from Nesterov (2004, Lemma 2.2.4). In our notation, we have

$$\frac{1}{\sqrt{\theta_{k+1}}} - \frac{1}{\sqrt{\theta_k}} = \frac{\theta_k - \theta_{k+1}}{\sqrt{\theta_k \theta_{k+1}}(\sqrt{\theta_k} + \sqrt{\theta_{k+1}})} \geq \frac{\theta_k(1 - (1 - 1/t_{k+1}))}{2\theta_k \sqrt{\theta_{k+1}}}$$

since θ_k is non-increasing. It follows that

$$\frac{1}{\sqrt{\theta_{k+1}}} - \frac{1}{\sqrt{\theta_k}} \geq \frac{1}{2t_{k+1}\sqrt{\theta_{k+1}}} = \frac{1}{2} \frac{1}{\sqrt{\prod_{n=0}^{k} \omega_n}} \geq \frac{1}{2},$$

showing that

$$1/\sqrt{\theta_k} \geq \frac{k-1}{2} + t_1/\sqrt{\omega_0} \geq \frac{k+1}{2}.$$

Hence, provided that $\sqrt{q}t_0 \leq 1$, we also find

$$\theta_k \leq \frac{4}{(k+1)^2}. \tag{B.10}$$

We have shown the following result.

Theorem B.1. If $\sqrt{q}t_0 \leq 1$, $t_0 \geq 0$, then the sequence (x^k) produced by iterations $x^k = T_\tau y^k$ with (B.9), (B.3), (B.5), (B.6) satisfies

$$F(x^k) - F(x^*) \leq r_k(q)\left(t_0^2(F(x^0) - F(x^*)) + \frac{1 + \tau\mu_g}{2\tau}\|x^0 - x^*\|^2\right), \tag{B.11}$$

where x^* is a minimizer of F, and

$$r_k(q) = \begin{cases} \min\left\{\dfrac{(1-\sqrt{q})^k}{t_0^2}, \dfrac{4}{(k+1)^2}\right\} & \text{if } t_0 \geq 1, \\ \min\left\{(1+\sqrt{q})(1-\sqrt{q})^k, \dfrac{4}{(k+1)^2}\right\} & \text{if } t_0 \in [0,1). \end{cases}$$

Theorem 4.10 is a particular case of this result, for $t_0 = 0$. □

Remark B.2 (constant steps). If $\mu > 0$ (which is $q > 0$), then an admissible choice which satisfies (B.3), (B.4) and (B.5) is to take $t = 1/\sqrt{q}$,

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

$\omega = 1 - \sqrt{q}$, and

$$\beta = \omega^2 \frac{1+\tau\mu_g}{1-\tau\mu_f} = \frac{\sqrt{1+\tau\mu_g} - \sqrt{\tau\mu}}{\sqrt{1+\tau\mu_g} + \sqrt{\tau\mu}}.$$

Then (B.11) becomes

$$F(x^k) - F(x^*) \le (1-\sqrt{q})^k \left(F(x^0) - F(x^*) + \mu \frac{\|x^0 - x^*\|^2}{2} \right).$$

Remark B.3 (monotone algorithms). The algorithms studied here are not necessarily 'monotone' in the sense that the objective F is not always non-increasing. A workaround implemented in various papers (Tseng 2008, Beck and Teboulle 2009) consists in choosing x^{k+1} to be *any* point for which $F(x^{k+1}) \le F(T_\tau y^k)$,[27] which will not change (B.1) much except that, in the last term, x^{k+1} should be replaced with $T_\tau y^k$. Then, the same computations carry on, and it is enough to replace the update rule (B.6) for y^k with

$$\begin{aligned} y^k &= x^k + \beta_k (x^k - x^{k-1}) + \omega_k \frac{t_k}{t_{k+1}} \frac{1+\tau\mu_g}{1-\tau\mu_f} (T_\tau y^{k-1} - x^k) \\ &= x^k + \beta_k \left((x^k - x^{k-1}) + \frac{t_k}{t_k - 1} (T_\tau y^{k-1} - x^k) \right) \end{aligned} \tag{B.6$'$}$$

to obtain the same rates of convergence. The most sensible choice for x^{k+1} is to take $T_\tau y^k$ if $F(T_\tau y^k) \le F(x^k)$, and x^k otherwise (see 'MFISTA' in Beck and Teboulle 2009), in which case one of the two terms ($x^k - x^{k-1}$ or $T_\tau y^{k-1} - x^k$) vanishes in (B.6$'$).

Tao *et al.* (2015) recently suggested choosing x^{k+1} to be the point reaching the minimum value between $F(T_\tau y^k)$ and $F(T_\tau x^k)$ (this requires additional computation), hoping to attain the best rate of accelerated and non-accelerated proximal descents, and thus obtain a linear convergence rate for the standard 'FISTA' ($\mu = 0$) implementation if F turns out to be strongly convex. This is very reasonable and seems to be supported by experiment, but we are not sure how to prove it.

C. Convergence rates for primal–dual algorithms

The goal of this appendix is to give an idea of how the convergence results in Section 5.1 are established, and also to explicitly give a proof of a variant of the accelerated rate in Theorem 5.2, which to our knowledge is not found in the current literature, and is an easy adaption of the proofs in Chambolle and Pock (2011, 2015*a*). In Section C.1 we sketch a proof of Theorem 5.1,

[27] This makes sense only if the evaluation of F is easy and does not take too much time.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

establishing the $O(1/k)$ ergodic convergence rate, while in Section C.2 we prove a variant of Theorem 5.2.

C.1. The PDHG and Condat–Vũ algorithm

We give an elementary proof of the rate of convergence of Theorem 5.1. It is easily extended to the non-linear proximity operator (or 'mirror descent': Beck and Teboulle 2003); in fact the proof is identical (Chambolle and Pock 2015*a*).

The first observation is that if $(\hat{x}, \hat{y}) = \mathcal{PD}_{\tau,\sigma}(\bar{x}, \bar{y}, \tilde{x}, \tilde{y})$, then for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$ we have

$$\begin{aligned}
&\mathcal{L}(\hat{x}, y) - \mathcal{L}(x, \hat{y}) \\
&\quad \le \frac{1}{2\tau}\|x - \bar{x}\|^2 - \frac{1 + \tau\mu_g}{2\tau}\|x - \hat{x}\|^2 - \frac{1 - L_h\tau}{2\tau}\|\hat{x} - \bar{x}\|^2 \\
&\qquad + \frac{1}{2\sigma}\|y - \bar{y}\|^2 - \frac{1}{2\sigma}\|y - \hat{y}\|^2 - \frac{1}{2\sigma}\|\hat{y} - \bar{y}\|^2 \\
&\qquad + \langle K(x - \hat{x}), \tilde{y} - \hat{y}\rangle - \langle K(\tilde{x} - \hat{x}), y - \hat{y}\rangle. \qquad \text{(C.1)}
\end{aligned}$$

In fact this follows from (4.36), applied on one hand to the function $g(x) + h(x) + \langle Kx, \tilde{y}\rangle$, and on the other hand to $f^*(y) - \langle K\tilde{x}, y\rangle$. For all x, y, we obtain

$$\begin{aligned}
&g(x) + h(x) + \langle Kx, \tilde{y}\rangle + \frac{1}{2\tau}\|x - \bar{x}\|^2 \\
&\qquad \ge g(\hat{x}) + h(\hat{x}) + \langle K\hat{x}, \tilde{y}\rangle + \frac{1 - \tau L_h}{2\tau}\|\bar{x} - \hat{x}\|^2 + \frac{1 + \tau\mu_g}{2\tau}\|x - \hat{x}\|^2, \\
&f^*(y) - \langle K\tilde{x}, y\rangle + \frac{1}{2\sigma}\|y - \bar{y}\|^2 \\
&\qquad \ge f^*(\hat{y}) - \langle K\tilde{x}, \hat{y}\rangle + \frac{1}{2\sigma}\|\bar{y} - \hat{y}\|^2 + \frac{1}{2\sigma}\|\hat{y} - y\|^2,
\end{aligned}$$

where $\mu_g \ge 0$ is a convexity parameter for g, which we will consider in Section C.2. Summing these two inequalities and rearranging, we obtain (C.1).

The PDHG algorithm corresponds to the choice $(\tilde{x}, \tilde{y}) = (2x^{k+1} - x^k, y^k)$, $(\hat{x}, \hat{y}) = (x^{k+1}, y^{k+1})$, $(\bar{x}, \bar{y}) = (x^k, y^k)$. We deduce (assuming $\mu_g = 0$) that

$$\begin{aligned}
&\mathcal{L}(x^{k+1}, y) - \mathcal{L}(x, y^{k+1}) \\
&\quad + \frac{1}{2\tau}\|x - x^{k+1}\|^2 + \frac{1}{2\sigma}\|y - y^{k+1}\|^2 - \langle K(x - x^{k+1}), y - y^{k+1}\rangle \\
&\quad + \frac{1 - \tau L_h}{2\tau}\|x^{k+1} - x^k\|^2 + \frac{1}{2\sigma}\|y^{k+1} - y^k\|^2 - \langle K(x^{k+1} - x^k), y^{k+1} - y^k\rangle \\
&\le \frac{1}{2\tau}\|x - x^k\|^2 + \frac{1}{2\sigma}\|y - y^k\|^2 - \langle K(x - x^k), y - y^k\rangle.
\end{aligned}$$

Thanks to (5.9), each of the last three lines is non-negative. We sum this

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

from $i = 0, \dots, k-1$, and find that

$$\sum_{i=1}^{K} \mathcal{L}(x^i, y) - \mathcal{L}(x, y^i) \leq \frac{1}{2\tau}\|x - x^0\|^2 + \frac{1}{2\sigma}\|y - y^0\|^2 - \langle K(x - x^0), y - y^0\rangle.$$

Equation (5.10) follows from the convexity of $(\xi, \eta) \mapsto \mathcal{L}(\xi, y) - \mathcal{L}(x, \eta)$, and using

$$2\langle K(x - x^0), y - y^0\rangle \leq \frac{\|x - x^0\|^2}{\tau} + \frac{\|y - y^0\|^2}{\sigma}.$$

C.2. An accelerated primal–dual algorithm

We briefly show how one can derive a result similar to Theorem 5.2 for a variant of the algorithm, which consists in performing the over-relaxation step in the variable y rather than the variable x. We still consider problem (5.8), in the case where g is μ_g-convex with $\mu_g > 0$, and as before ∇h is L_h-Lipschitz. Given $x^0, y^0 = y^{-1}$, we let at each iteration

$$(x^{k+1}, y^{k+1}) = \mathcal{PD}_{\tau_k, \sigma_k}(x^k, y^k, x^{k+1}, y^k + \theta_k(y^k - y^{k-1})),$$

where $\theta_k, \tau_k, \sigma_k$ will be made precise later. We let

$$x^{k+1} = \operatorname{prox}_{\tau g}(x^k - \tau(\nabla h(x^k) + K^*(y^k + \theta_k(y^k - y^{k-1})))), \tag{C.2}$$

$$y^{k+1} = \operatorname{prox}_{\sigma f^*}(y^k + \sigma K x^{k+1}). \tag{C.3}$$

Using (C.1) with $(\tilde{x}, \tilde{y}) = (x^{k+1}, y^k + \theta_k(y^k - y^{k-1}))$, we obtain that for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$,

$$\begin{aligned}
&\frac{1}{2\tau_k}\|x - x^k\|^2 + \frac{1}{2\sigma_k}\|y - y^k\|^2 \\
&\quad \geq \mathcal{L}(x^{k+1}, y) - \mathcal{L}(x, y^{k+1}) + \frac{1 + \tau_k\mu_g}{2\tau_k}\|x - x^{k+1}\|^2 + \frac{1}{2\sigma_k}\|y - y^{k+1}\|^2 \\
&\qquad - \langle K(x^{k+1} - x), y^{k+1} - y^k\rangle + \theta_k\langle K(x^{k+1} - x), y^k - y^{k-1}\rangle \\
&\qquad + \frac{1 - \tau_k L_h}{2\tau_k}\|x^k - x^{k+1}\|^2 + \frac{1}{2\sigma_k}\|y^k - y^{k+1}\|^2.
\end{aligned}$$

Letting

$$\Delta_k(x, y) := \frac{\|x - x^k\|^2}{2\tau_k} + \frac{\|y - y^k\|^2}{2\sigma_k},$$

and assuming we can choose $(\tau_k, \sigma_k, \theta_k)$ satisfying

$$\frac{1 + \tau_k\mu_g}{\tau_k} \geq \frac{1}{\theta_{k+1}\tau_{k+1}}, \tag{C.4}$$

$$\sigma_k = \theta_{k+1}\sigma_{k+1}, \tag{C.5}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

we obtain

$$\begin{aligned}\Delta_k(x,y) &- \theta_k\langle K(x^k - x), y^k - y^{k-1}\rangle \\ &\geq \mathcal{L}(x^{k+1}, y) - \mathcal{L}(x, y^{k+1}) + \frac{1}{\theta_{k+1}}\Delta_{k+1}(x,y) \\ &\quad - \langle K(x^{k+1} - x), y^{k+1} - y^k\rangle + \frac{1 - \tau_k L_h}{2\tau_k}\|x^{k+1} - x^k\|^2 \\ &\quad + \frac{1}{2\sigma_k}\|y^{k+1} - y^k\|^2 + \theta_k\langle K(x^{k+1} - x^k), y^k - y^{k-1}\rangle.\end{aligned}$$

Now using (with $L = \|K\|$)

$$\begin{aligned}&\theta_k\langle K(x^{k+1} - x^k), y^k - y^{k-1}\rangle \\ &\quad \geq -\frac{\theta_k^2 L^2 \sigma_k \tau_k}{2\tau_k}\|x^k - x^{k+1}\|^2 - \frac{1}{2\sigma_k}\|y^k - y^{k-1}\|^2,\end{aligned}$$

we find, using $1/\theta_{k+1} = \sigma_{k+1}/\sigma_k$, that

$$\begin{aligned}\Delta_k(x,y) &- \theta_k\langle K(x^k - x), y^k - y^{k-1}\rangle + \frac{1}{2\sigma_k}\|y^k - y^{k-1}\|^2 \\ &\geq \mathcal{L}(x^{k+1}, y) - \mathcal{L}(x, y^{k+1}) \\ &\quad + \frac{\sigma_{k+1}}{\sigma_k}\Big(\Delta_{k+1}(x,y) - \theta_{k+1}\langle K(x^{k+1} - x), y^{k+1} - y^k\rangle \\ &\qquad\qquad + \frac{1}{2\sigma_{k+1}}\|y^{k+1} - y^k\|^2\Big) \\ &\quad + \frac{1 - \tau_k L_h - \theta_k^2 L^2 \sigma_k \tau_k}{2\tau_k}\|x^{k+1} - x^k\|^2.\end{aligned}$$

If we can ensure, for all k, that

$$\tau_k L_h + \theta_k^2 L^2 \sigma_k \tau_k \leq 1, \tag{C.6}$$

then we will deduce by induction that

$$\begin{aligned}\Delta_0(x,y) &\geq \sum_{i=1}^{k} \frac{\sigma_{i-1}}{\sigma_0}\big(\mathcal{L}(x^i, y) - \mathcal{L}(x, y^i)\big) \\ &\quad + \frac{\sigma_k}{\sigma_0}\left(\Delta_k(x,y) - \theta_k\langle K(x^k - x), y^k - y^{k-1}\rangle + \frac{1}{2\sigma_k}\|y^k - y^{k-1}\|^2\right).\end{aligned}$$

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Finally, letting

$$T_k = \sum_{i=1}^{k} \frac{\sigma_{i-1}}{\sigma_0}, \quad (X^k, Y^k) = \frac{1}{T_k} \sum_{i=1}^{k} \frac{\sigma_{i-1}}{\sigma_0} (x^k, y^k), \tag{C.7}$$

for each k and using the convexity of $\mathcal{L}(\cdot, y) - \mathcal{L}(x, \cdot)$, we find that

$$\begin{aligned}\Delta_0(x,y) \geq T_k\big(\mathcal{L}(X^k, y) - \mathcal{L}(x, Y^k)\big) \\ + \frac{\sigma_k}{\sigma_0} \frac{1 - \theta_k^2 L^2 \tau_k \sigma_k}{2\tau_k} \|x_k - x\|^2 + \frac{1}{2\sigma_0} \|y_k - y\|^2.\end{aligned}$$

There are several choices of $\tau_k, \sigma_k, \theta_k$ that will ensure a good rate of convergence for the ergodic gap or for the distance $\|x_k - x\|$; see Chambolle and Pock (2015*a*) for a discussion. A simple choice, as in Chambolle and Pock (2011), is to take, for $k \geq 0$,

$$\theta_{k+1} = \frac{1}{\sqrt{1 + \mu_g \tau_k}}, \tag{C.8}$$

$$\tau_{k+1} = \theta_{k+1} \tau_k, \quad \sigma_{k+1} = \frac{\sigma_k}{\theta_{k+1}}. \tag{C.9}$$

One can show that in this case, since

$$\frac{1}{\tau_{k+1}} = \frac{1}{\tau_k} + \frac{\mu_g}{1 + \sqrt{1 + \mu_g \tau_k}},$$

we have $\tau_k \sim 2/(\mu_g k)$, so that $1/T_k = O(1/k^2)$. In this case,

$$\tau_k L_h + \theta_k^2 L^2 \sigma_k \tau_k = \tau_k L_h + \theta_k^2 L^2 \sigma 0 \tau_0 \leq \tau_0 (L_h + L^2 \sigma_0),$$

so choosing σ_0 arbitrarily and $\tau_0 \leq 1/(L_h + L^2 \sigma_0)$ will yield convergence. If, moreover, $\tau_0 (L_h + L^2 \sigma_0) = t < 1$, we find, in addition to the ergodic rate

$$\mathcal{L}(X^k, y) - \mathcal{L}(x, Y^k) \leq \frac{1}{T_k} \Delta_0(x, y) \tag{C.10}$$

for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$, the rate for the iterate x^k:

$$\|x_k - x^*\|^2 \lesssim \frac{4}{(1-t)\mu_g^2 k^2} \left(\frac{1}{\tau_0^2} \|x^0 - x^*\|^2 + \frac{1}{\sigma_0 \tau_0} \|y^0 - y^*\|^2 \right), \tag{C.11}$$

where (x^*, y^*) is a saddle point.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

REFERENCES[28]

M. Aharon, M. Elad and A. Bruckstein (2006), 'K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation', *IEEE Trans. Signal Process.* **54**, 4311–4322.

R. K. Ahuja, T. L. Magnanti and J. B. Orlin (1993), *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall.

M. A. Aĭzerman, È. M. Braverman and L. I. Rozonoèr (1964), 'A probabilistic problem on automata learning by pattern recognition and the method of potential functions', *Avtomat. i Telemeh.* **25**, 1307–1323.

G. Alberti, G. Bouchitté and G. Dal Maso (2003), 'The calibration method for the Mumford–Shah functional and free-discontinuity problems', *Calc. Var. Partial Differential Equations* **16**, 299–333.

Z. Allen-Zhu and L. Orecchia (2014), Linear coupling: An ultimate unification of gradient and mirror descent. arXiv:1407.1537

M. Almeida and M. A. T. Figueiredo (2013), 'Deconvolving images with unknown boundaries using the alternating direction method of multipliers', *IEEE Trans. Image Process.* **22**, 3074–3086.

F. Alvarez (2003), 'Weak convergence of a relaxed and inertial hybrid projection-proximal point algorithm for maximal monotone operators in Hilbert space', *SIAM J. Optim.* **14**, 773–782.

F. Alvarez and H. Attouch (2001), 'An inertial proximal method for maximal monotone operators via discretization of a nonlinear oscillator with damping', *Set-Valued Analysis* **9**, 3–11.

L. Ambrosio and S. Masnou (2003), 'A direct variational approach to a problem arising in image reconstruction', *Interfaces Free Bound.* **5**, 63–81.

L. Ambrosio and V. M. Tortorelli (1992), 'On the approximation of free discontinuity problems', *Boll. Un. Mat. Ital. B* (7) **6**, 105–123.

L. Ambrosio, N. Fusco and D. Pallara (2000), *Functions of Bounded Variation and Free Discontinuity Problems*, The Clarendon Press, Oxford University Press.

P. Arias, G. Facciolo, V. Caselles and G. Sapiro (2011), 'A variational framework for exemplar-based image inpainting', *Internat. J. Computer Vision* **93**, 319–347.

L. Armijo (1966), 'Minimization of functions having Lipschitz continuous first partial derivatives', *Pacific J. Math.* **16**, 1–3.

K. J. Arrow, L. Hurwicz and H. Uzawa (1958), *Studies in Linear and Non-linear Programming*, Vol. II of *Stanford Mathematical Studies in the Social Sciences*, Stanford University Press.

A. d'Aspremont (2008), 'Smooth optimization with approximate gradient', *SIAM J. Optim.* **19**, 1171–1183.

H. Attouch, J. Bolte and B. F. Svaiter (2013), 'Convergence of descent methods for semi-algebraic and tame problems: Proximal algorithms, forward–backward splitting, and regularized Gauss–Seidel methods', *Math. Program.* A **137**, 91–129.

[28] The URLs cited in this work were correct at the time of going to press, but the publisher and the authors make no undertaking that the citations remain live or are accurate or appropriate.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

H. Attouch, J. Bolte, P. Redont and A. Soubeyran (2010), 'Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka–Łojasiewicz inequality', *Math. Oper. Res.* **35**, 438–457.

H. Attouch, L. M. Briceño-Arias and P. L. Combettes (2009/10), 'A parallel splitting method for coupled monotone inclusions', *SIAM J. Control Optim.* **48**, 3246–3270.

H. Attouch, G. Buttazzo and G. Michaille (2014), *Variational Analysis in Sobolev and BV Spaces: Applications to PDEs and Optimization*, second edition, MOS–SIAM Series on Optimization, SIAM.

J.-F. Aujol and C. Dossal (2015), 'Stability of over-relaxations for the forward–backward algorithm, application to FISTA', *SIAM J. Optim.* **25**, 2408–2433.

A. Auslender (1976), *Optimisation: Méthodes Numériques*, Maîtrise de Mathématiques et Applications Fondamentales, Masson.

A. Auslender and M. Teboulle (2004), 'Interior gradient and epsilon-subgradient descent methods for constrained convex minimization', *Math. Oper. Res.* **29**, 1–26.

H. H. Bauschke and P. L. Combettes (2011), *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC, Springer.

H. H. Bauschke, S. M. Moffat and X. Wang (2012), 'Firmly nonexpansive mappings and maximally monotone operators: Correspondence and duality', *Set-Valued Var. Anal.* **20**, 131–153.

A. Beck (2015), 'On the convergence of alternating minimization for convex programming with applications to iteratively reweighted least squares and decomposition schemes', *SIAM J. Optim.* **25**, 185–209.

A. Beck and M. Teboulle (2003), 'Mirror descent and nonlinear projected subgradient methods for convex optimization', *Oper. Res. Lett.* **31**, 167–175.

A. Beck and M. Teboulle (2009), 'A fast iterative shrinkage-thresholding algorithm for linear inverse problems', *SIAM J. Imaging Sci.* **2**, 183–202.

A. Beck and L. Tetruashvili (2013), 'On the convergence of block coordinate descent type methods', *SIAM J. Optim.* **23**, 2037–2060.

S. Becker and P. L. Combettes (2014), 'An algorithm for splitting parallel sums of linearly composed monotone operators, with applications to signal recovery', *J. Nonlinear Convex Anal.* **15**, 137–159.

S. Becker and J. Fadili (2012), A quasi-Newton proximal splitting method. In *Advances in Neural Information Processing Systems 25*, pp. 2627–2635.

S. Becker, J. Bobin and E. Candès (2011), 'NESTA: A fast and accurate first-order method for sparse recovery', *SIAM J. Imaging Sci.* **4**, 1–39.

J. Bect, L. Blanc-Féraud, G. Aubert and A. Chambolle (2004), A l^1-unified framework for image restoration. In *Proc. 8th European Conference on Computer Vision: ECCV 2004*, Part IV (T. Pajdla and J. Matas, eds), Vol. 2034 of *Lecture Notes in Computer Science*, Springer, pp. 1–13.

A. Ben-Tal and A. Nemirovski (1998), 'Robust convex optimization', *Math. Oper. Res.* **23**, 769–805.

A. Ben-Tal and A. Nemirovski (2001), *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*, MPS/SIAM Series on Optimization, SIAM.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

A. Ben-Tal, L. El Ghaoui and A. Nemirovski (2009), *Robust Optimization*, Princeton Series in Applied Mathematics, Princeton University Press.
J.-D. Benamou, G. Carlier, M. Cuturi, L. Nenna and G. Peyré (2015), 'Iterative Bregman projections for regularized transportation problems', *SIAM J. Sci. Comput.* **37**, A1111–A1138.
A. Benfenati and V. Ruggiero (2013), 'Inexact Bregman iteration with an application to Poisson data reconstruction', *Inverse Problems* **29**, 065016.
D. P. Bertsekas (2015), *Convex Optimization Algorithms*, Athena Scientific.
D. P. Bertsekas and S. K. Mitter (1973), 'Descent numerical methods for optimization problems with nondifferentiable cost functions', *SIAM J. Control* **11**, 637–652.
J. Bioucas-Dias and M. Figueiredo (2007), 'A new TwIST: Two-step iterative shrinkage/thresholding algorithms for image restoration', *IEEE Trans. Image Process.* **16**, 2992–3004.
A. Blake and A. Zisserman (1987), *Visual Reconstruction*, MIT Press.
P. Blomgren and T. F. Chan (1998), 'Color TV: total variation methods for restoration of vector-valued images', *IEEE Trans. Image Process.* **7**, 304–309.
J. Bolte, A. Daniilidis and A. Lewis (2006), 'The Łojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems', *SIAM J. Optim.* **17**, 1205–1223.
J. Bolte, S. Sabach and M. Teboulle (2014), 'Proximal alternating linearized minimization for nonconvex and nonsmooth problems', *Math. Program.* A **146**, 459–494.
S. Bonettini and V. Ruggiero (2012), 'On the convergence of primal–dual hybrid gradient algorithms for total variation image restoration', *J. Math. Imaging Vision* **44**, 236–253.
S. Bonettini, A. Benfenati and V. Ruggiero (2014), Scaling techniques for ϵ-subgradient projection methods. arXiv:1407.6133
S. Bonettini, F. Porta and V. Ruggiero (2015), A variable metric forward–backward method with extrapolation. arXiv:1506.02900
J. F. Bonnans, J. C. Gilbert, C. Lemaréchal and C. A. Sagastizábal (1995), 'A family of variable metric proximal methods', *Math. Program.* A **68**, 15–47.
A. Bonnet and G. David (2001), 'Cracktip is a global Mumford–Shah minimizer', *Astérisque*, Sociéte Mathématique de France, Vol. 274.
J. Borwein and D. Luke (2015), Duality and convex programming. In *Handbook of Mathematical Methods in Imaging* (O. Scherzer, ed.), Springer, pp. 257–304.
R. I. Boţ, E. R. Csetnek, A. Heinrich and C. Hendrich (2015), 'On the convergence rate improvement of a primal–dual splitting algorithm for solving monotone inclusion problems', *Math. Program.* A **150**, 251–279.
G. Bouchitté (2006), Convex analysis and duality methods. In *Encyclopedia of Mathematical Physics* (J.-P. Françoise, G. L. Naber and T. S. Tsun, eds), Academic, pp. 642–652.
S. Boyd and L. Vandenberghe (2004), *Convex Optimization*, Cambridge University Press.
S. Boyd, N. Parikh, E. Chu, B. Peleato and J. Eckstein (2011), 'Distributed optimization and statistical learning via the alternating direction method of multipliers', *Found. Trends Mach. Learn.* **3**, 1–122.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

Y. Boykov and V. Kolmogorov (2004), 'An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision', *IEEE Trans. PAMI* **26**, 1124–1137.

Y. Boykov, O. Veksler and R. Zabih (2001), 'Fast approximate energy minimization via graph cuts', *IEEE Trans. PAMI* **23**, 1222–1239.

K. A. Brakke (1995), 'Soap films and covering spaces', *J. Geom. Anal.* **5**, 445–514.

K. Bredies and D. A. Lorenz (2008), 'Linear convergence of iterative soft-thresholding', *J. Fourier Anal. Appl.* **14**, 813–837.

K. Bredies and H. Sun (2015*a*), Preconditioned alternating direction method of multipliers for the minimization of quadratic plus non-smooth convex functionals. Technical report, University of Graz.

K. Bredies and H. Sun (2015*b*), 'Preconditioned Douglas–Rachford splitting methods for convex-concave saddle-point problems', *SIAM J. Numer. Anal.* **53**, 421–444.

K. Bredies and H. P. Sun (2015*c*), 'Preconditioned Douglas–Rachford algorithms for TV- and TGV-regularized variational imaging problems', *J. Math. Imaging Vision* **52**, 317–344.

K. Bredies, K. Kunisch and T. Pock (2010), 'Total generalized variation', *SIAM J. Imaging Sci.* **3**, 492–526.

K. Bredies, D. A. Lorenz and S. Reiterer (2015*a*), 'Minimization of non-smooth, non-convex functionals by iterative thresholding', *J. Optim. Theory Appl.* **165**, 78–112.

K. Bredies, T. Pock and B. Wirth (2013), 'Convex relaxation of a class of vertex penalizing functionals', *J. Math. Imaging Vision* **47**, 278–302.

K. Bredies, T. Pock and B. Wirth (2015*b*), 'A convex, lower semicontinuous approximation of Euler's elastica energy', *SIAM J. Math. Anal.* **47**, 566–613.

L. M. Bregman [Brègman] (1967), 'A relaxation method of finding a common point of convex sets and its application to the solution of problems in convex programming', *Ž. Vyčisl. Mat. i Mat. Fiz.* **7**, 620–631.

X. Bresson and T. F. Chan (2008), 'Fast dual minimization of the vectorial total variation norm and applications to color image processing', *Inverse Probl. Imaging* **2**, 455–484.

H. Brézis (1973), *Opérateurs Maximaux Monotones et Semi-Groupes de Contractions dans les Espaces de Hilbert*, Vol. 5 of *North-Holland Mathematics Studies*, North-Holland.

H. Brézis (1983), *Analyse Fonctionnelle: Théorie et Applications*, Collection Mathématiques Appliquées pour la Maîtrise, Masson.

H. Brézis and P.-L. Lions (1978), 'Produits infinis de résolvantes', *Israel J. Math.* **29**, 329–345.

L. M. Briceño-Arias and P. L. Combettes (2011), 'A monotone + skew splitting model for composite monotone inclusions in duality', *SIAM J. Optim.* **21**, 1230–1250.

F. E. Browder (1967), 'Convergence theorems for sequences of nonlinear operators in Banach spaces', *Math. Z.* **100**, 201–225.

F. E. Browder and W. V. Petryshyn (1966), 'The solution by iteration of nonlinear functional equations in Banach spaces', *Bull. Amer. Math. Soc.* **72**, 571–575.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

T. Brox, A. Bruhn, N. Papenberg and J. Weickert (2004), High accuracy optical flow estimation based on a theory for warping. In *Proc. 8th European Conference on Computer Vision: ECCV 2004*, Part IV (T. Pajdla and J. Matas, eds), Vol. 2034 of *Lecture Notes In Computer Science*, Springer, pp. 25–36.

J. Bruna and S. Mallat (2013), 'Invariant scattering convolution networks', *IEEE Trans. PAMI* **35**, 1872–1886.

J. Bruna, A. Szlam and Y. LeCun (2014), Signal recovery from pooling representations. In *31st International Conference on Machine Learning, ICML 2014*, Vol. 2, International Machine Learning Society (IMLS), pp. 1585–1598.

A. Buades, B. Coll and J. M. Morel (2005), 'A review of image denoising algorithms, with a new one', *Multiscale Model. Simul.* **4**, 490–530.

A. Buades, B. Coll and J. M. Morel (2011), 'Non-local means denoising', *Image Processing On Line* **1**.

M. Burger, A. Sawatzky and G. Steidl (2014), First order algorithms in variational image processing. arXiv:1412.4237. To appear in *Splitting Methods in Communication and Imaging, Science and Engineering* (R. Glowinski, S. J. Osher and W. Yin, eds), Springer.

C. J. C. Burges (1998), 'A tutorial on support vector machines for pattern recognition', *Data Min. Knowl. Discov.* **2**, 121–167.

J. V. Burke and M. Qian (1999), 'A variable metric proximal point algorithm for monotone operators', *SIAM J. Control Optim.* **37**, 353–375.

J. V. Burke and M. Qian (2000), 'On the superlinear convergence of the variable metric proximal point algorithm using Broyden and BFGS matrix secant updating', *Math. Program.* A **88**, 157–181.

R. H. Byrd, P. Lu, J. Nocedal and C. Y. Zhu (1995), 'A limited memory algorithm for bound constrained optimization', *SIAM J. Sci. Comput.* **16**, 1190–1208.

J.-F. Cai, E. J. Candès and Z. Shen (2010), 'A singular value thresholding algorithm for matrix completion', *SIAM J. Optim.* **20**, 1956–1982.

E. Candès, L. Demanet, D. Donoho and L. Ying (2006*a*), 'Fast discrete curvelet transforms', *Multiscale Model. Simul.* **5**, 861–899.

E. J. Candès, X. Li, Y. Ma and J. Wright (2011), 'Robust principal component analysis?', *J. Assoc. Comput. Mach.* **58**, #11.

E. J. Candès, J. Romberg and T. Tao (2006*b*), 'Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information', *IEEE Trans. Inform. Theory* pp. 489–509.

A. Chambolle (1994), Partial differential equations and image processing. In *Proc. ICIP-94: 1994 IEEE International Conference on Image Processing*, pp. 16–20.

A. Chambolle (1999), 'Finite-differences discretizations of the Mumford–Shah functional', *M2AN Math. Model. Numer. Anal.* **33**, 261–288.

A. Chambolle (2004*a*), 'An algorithm for mean curvature motion', *Interfaces Free Bound.* **6**, 195–218.

A. Chambolle (2004*b*), 'An algorithm for total variation minimization and applications', *J. Math. Imaging Vision* **20**, 89–97.

A. Chambolle (2005), Total variation minimization and a class of binary MRF models. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Vol. 3757 of *Lecture Notes in Computer Science*, Springer, pp. 136–152.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

A. Chambolle and J. Darbon (2009), 'On total variation minimization and surface evolution using parametric maximum flows', *Internat. J. Comput. Vis.* **84**, 288–307.

A. Chambolle and J. Darbon (2012), A parametric maximum flow approach for discrete total variation regularization. In *Image Processing and Analysis with Graphs: Theory and Practice* (O. Lézoray and L. Grady, eds), CRC Press, pp. 93–109.

A. Chambolle and C. Dossal (2015), 'On the convergence of the iterates of the fast iterative shrinkage/thresholding algorithm', *J. Optim. Theory Appl.* **166**, 968–982.

A. Chambolle and P.-L. Lions (1995), Image restoration by constrained total variation minimization and variants. In *Investigative and Trial Image Processing. Proc. SPIE* **2567**, 50–59.

A. Chambolle and P.-L. Lions (1997), 'Image recovery via total variation minimization and related problems', *Numer. Math.* **76**, 167–188.

A. Chambolle and T. Pock (2011), 'A first-order primal–dual algorithm for convex problems with applications to imaging', *J. Math. Imaging Vision* **40**, 120–145.

A. Chambolle and T. Pock (2015*a*), 'On the ergodic convergence rates of a first-order primal–dual algorithm', *Math. Program.* A. doi:10.1007/s10107-015-0957-3

A. Chambolle and T. Pock (2015*b*), 'A remark on accelerated block coordinate descent for computing the proximity operators of a sum of convex functions', *SMAI J. Comput. Math.* **1**, 29–54.

A. Chambolle, D. Cremers and T. Pock (2012), 'A convex approach to minimal partitions', *SIAM J. Imaging Sci.* **5**, 1113–1158.

A. Chambolle, R. A. DeVore, N.-Y. Lee and B. J. Lucier (1998), 'Nonlinear wavelet image processing: Variational problems, compression, and noise removal through wavelet shrinkage', *IEEE Trans. Image Process.* **7**, 319–335.

A. Chambolle, S. E. Levine and B. J. Lucier (2011), 'An upwind finite-difference method for total variation-based image smoothing', *SIAM J. Imaging Sci.* **4**, 277–299.

T. F. Chan and S. Esedoḡlu (2005), 'Aspects of total variation regularized L^1 function approximation', *SIAM J. Appl. Math.* **65**, 1817–1837.

T. F. Chan and L. A. Vese (2001), 'Active contours without edges', *IEEE Trans. Image Process.* **10**, 266–277.

T. F. Chan and L. A. Vese (2002), Active contour and segmentation models using geometric PDE's for medical imaging. In *Geometric Methods in Bio-medical Image Processing*, Mathematics and Visualization, Springer, pp. 63–75.

T. F. Chan, S. Esedoḡlu and M. Nikolova (2006), 'Algorithms for finding global minimizers of image segmentation and denoising models', *SIAM J. Appl. Math.* **66**, 1632–1648.

T. F. Chan, G. H. Golub and P. Mulet (1999), 'A nonlinear primal–dual method for total variation-based image restoration', *SIAM J. Sci. Comput.* **20**, 1964–1977.

R. Chartrand and B. Wohlberg (2013), A nonconvex ADMM algorithm for group sparsity with sparse groups. In *Proc. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing: ICASSP*, IEEE, pp. 6009–6013.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

G. Chen and M. Teboulle (1993), 'Convergence analysis of a proximal-like minimization algorithm using Bregman functions', *SIAM J. Optim.* **3**, 538–543.

G. Chen and M. Teboulle (1994), 'A proximal-based decomposition method for convex minimization problems', *Math. Program.* A **64**, 81–101.

S. Chen and D. Donoho (1994), Basis pursuit. In *28th Asilomar Conference on Signals, Systems, and Computers*, IEEE, pp. 41–44.

S. S. Chen, D. L. Donoho and M. A. Saunders (1998), 'Atomic decomposition by basis pursuit', *SIAM J. Sci. Comput.* **20**, 33–61.

Y. Chen, G. Lan and Y. Ouyang (2014*a*), 'Optimal primal–dual methods for a class of saddle point problems', *SIAM J. Optim.* **24**, 1779–1814.

Y. Chen, R. Ranftl and T. Pock (2014*b*), 'Insights into analysis operator learning: From patch-based sparse models to higher order MRFs', *IEEE Trans. Image Process.* **23**, 1060–1072.

E. Chouzenoux, J.-C. Pesquet and A. Repetti (2014), 'Variable metric forward–backward algorithm for minimizing the sum of a differentiable function and a convex function', *J. Optim. Theory Appl.* **162**, 107–132.

E. Chouzenoux, J.-C. Pesquet and A. Repetti (2016), 'A block coordinate variable metric forward–backward algorithm', *J. Global Optim.* doi:10.1007/s10898-016-0405-9

D. C. Ciresan, U. Meier and J. Schmidhuber (2012), Multi-column deep neural networks for image classification. In *Proc. 25th IEEE Conference on Computer Vision and Pattern Recognition: CVPR 2012*, pp. 3642–3649.

G. Citti and A. Sarti (2006), 'A cortical based model of perceptual completion in the roto-translation space', *J. Math. Imaging Vision* **24**, 307–326.

P. L. Combettes (2004), 'Solving monotone inclusions via compositions of nonexpansive averaged operators', *Optimization* **53**, 475–504.

P. L. Combettes and J.-C. Pesquet (2011), Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, Vol. 49 of *Springer Optimization and its Applications*, Springer, pp. 185–212.

P. L. Combettes and B. C. Vũ (2014), 'Variable metric forward–backward splitting with applications to monotone inclusions in duality', *Optimization* **63**, 1289–1318.

P. L. Combettes and V. R. Wajs (2005), 'Signal recovery by proximal forward–backward splitting', *Multiscale Model. Simul.* **4**, 1168–1200.

L. Condat (2013*a*), 'A direct algorithm for 1D total variation denoising', *IEEE Signal Proc. Letters* **20**, 1054–1057.

L. Condat (2013*b*), 'A primal–dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms', *J. Optim. Theory Appl.* **158**, 460–479.

D. Cremers and E. Strekalovskiy (2013), 'Total cyclic variation and generalizations', *J. Math. Imaging Vision* **47**, 258–277.

K. Dabov, A. Foi, V. Katkovnik and K. Egiazarian (2007), 'Image denoising by sparse 3-D transform-domain collaborative filtering', *IEEE Trans. Image Process.* **16**, 2080–2095.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

G. Dal Maso, M. G. Mora and M. Morini (2000), 'Local calibrations for minimizers of the Mumford–Shah functional with rectilinear discontinuity sets', *J. Math. Pures Appl.* (9) **79**, 141–162.

A. Danielyan, V. Katkovnik and K. Egiazarian (2012), 'BM3D frames and variational image deblurring', *IEEE Trans. Image Process.* **21**, 1715–1728.

J. Darbon and M. Sigelle (2004), Exact optimization of discrete constrained total variation minimization problems. In *Combinatorial Image Analysis*, Vol. 3322 of *Lecture Notes in Computer Science*, Springer, pp. 548–557.

J. Darbon and M. Sigelle (2006*a*), 'Image restoration with discrete constrained total variation I: Fast and exact optimization', *J. Math. Imaging Vision* **26**, 261–276.

J. Darbon and M. Sigelle (2006*b*), 'Image restoration with discrete constrained total variation II: Levelable functions, convex priors and non-convex cases', *J. Math. Imaging Vision* **26**, 277–291.

I. Daubechies, M. Defrise and C. De Mol (2004), 'An iterative thresholding algorithm for linear inverse problems with a sparsity constraint', *Comm. Pure Appl. Math.* **57**, 1413–1457.

P. L. Davies and A. Kovac (2001), 'Local extremes, runs, strings and multiresolution', *Ann. Statist.* **29**, 1–65.

D. Davis (2015), 'Convergence rate analysis of primal–dual splitting schemes', *SIAM J. Optim.* **25**, 1912–1943.

D. Davis and W. Yin (2014*a*), Convergence rate analysis of several splitting schemes. arXiv:1406.4834

D. Davis and W. Yin (2014*b*), Faster convergence rates of relaxed Peaceman–Rachford and ADMM under regularity assumptions. arXiv:1407.5210

D. Davis and W. Yin (2015), A three-operator splitting scheme and its optimization applications. CAM Report 15-13, UCLA. arXiv:1504.01032

W. Deng and W. Yin (2016), 'On the global and linear convergence of the generalized alternating direction method of multipliers', *J. Sci. Comput.* **66**, 889–916.

R. A. DeVore (1998), Nonlinear approximation. In *Acta Numerica*, Vol. 7, Cambridge University Press, pp. 51–150.

D. L. Donoho (1995), 'De-noising by soft-thresholding', *IEEE Trans. Inform. Theory* **41**, 613–627.

D. L. Donoho (2006), 'Compressed sensing', *IEEE Trans. Inform. Theory* **52**, 1289–1306.

J. Douglas and H. H. Rachford (1956), 'On the numerical solution of heat conduction problems in two and three space variables', *Trans. Amer. Math. Soc.* **82**, 421–439.

Y. Drori, S. Sabach and M. Teboulle (2015), 'A simple algorithm for a class of nonsmooth convex-concave saddle-point problems', *Oper. Res. Lett.* **43**, 209–214.

K.-B. Duan and S. S. Keerthi (2005), Which is the best multiclass SVM method? An empirical study. In *Multiple Classifier Systems: Proc. 6th International Workshop, MCS 2005* (N. C. Oza, R. Polikar, J. Kittler and F. Roli, eds), Vol. 3541 of *Lecture Notes in Computer Science*, Springer, pp. 278–285.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

J. Duchi, S. Shalev-Shwartz, Y. Singer and T. Chandra (2008), Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proc. 25th International Conference on Machine Learning: ICML '08*, ACM, pp. 272–279.

F.-X. Dupé, M. J. Fadili and J.-L. Starck (2012), 'Deconvolution under Poisson noise using exact data fidelity and synthesis or analysis sparsity priors', *Statist. Methodol.* **9**, 4–18.

J. Duran, M. Moeller, C. Sbert and D. Cremers (2016*a*), 'On the implementation of collaborative TV regularization: Application to cartoon+texture decomposition', *Image Processing On Line* **6**, 27–74.

J. Duran, M. Möller, C. Sbert and D. Cremers (2016*b*), 'Collaborative total variation: A general framework for vectorial TV models', *SIAM J. Imaging Sci.* **9**, 116–151.

R. L. Dykstra (1983), 'An algorithm for restricted least squares regression', *J. Amer. Statist. Assoc.* **78**(384), 837–842.

G. Easley, D. Labate and W.-Q. Lim (2008), 'Sparse directional image representations using the discrete shearlet transform', *Appl. Comput. Harmon. Anal.* **25**, 25–46.

J. Eckstein (1989), Splitting methods for monotone operators with applications to parallel optimization. PhD thesis, Massachusetts Institute of Technology.

J. Eckstein (1993), 'Nonlinear proximal point algorithms using Bregman functions, with applications to convex programming', *Math. Oper. Res.* **18**, 202–226.

J. Eckstein and D. P. Bertsekas (1992), 'On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators', *Math. Program.* A **55**, 293–318.

I. Ekeland and R. Témam (1999), *Convex Analysis and Variational Problems* (translated from French), Vol. 28 of *Classics in Applied Mathematics*, SIAM.

E. Esser (2009), Applications of Lagrangian-based alternating direction methods and connections to split Bregman. CAM Report 09-31, UCLA.

E. Esser, X. Zhang and T. F. Chan (2010), 'A general framework for a class of first order primal–dual algorithms for convex optimization in imaging science', *SIAM J. Imaging Sci.* **3**, 1015–1046.

L. C. Evans and R. F. Gariepy (1992), *Measure Theory and Fine Properties of Functions*, CRC Press.

H. Federer (1969), *Geometric Measure Theory*, Springer.

O. Fercoq and P. Bianchi (2015), A coordinate descent primal–dual algorithm with large step size and possibly non separable functions. arXiv:1508.04625

O. Fercoq and P. Richtárik (2013), Smooth minimization of nonsmooth functions with parallel coordinate descent methods. arXiv:1309.5885

O. Fercoq and P. Richtárik (2015), 'Accelerated, parallel and proximal coordinate descent', *SIAM J. Optim.* **25**, 1997–2023.

S. Ferradans, N. Papadakis, G. Peyré and J.-F. Aujol (2014), 'Regularized discrete optimal transport', *SIAM J. Imaging Sci.* **7**, 1853–1882.

M. Fortin and R. Glowinski (1982), *Méthodes de Lagrangien Augmenté: Applications à la Résolution Numérique de Problèmes aux Limites*, Vol. 9 of *Méthodes Mathématiques de l'Informatique*, Gauthier-Villars.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

X. L. Fu, B. S. He, X. F. Wang and X. M. Yuan (2014), 'Block-wise alternating direction method of multipliers with Gaussian back substitution for multiple-block convex programming'.

M. Fukushima and H. Mine (1981), 'A generalized proximal point algorithm for certain nonconvex minimization problems', *Internat. J. Systems Sci.* **12**, 989–1000.

D. Gabay (1983), Applications of the method of multipliers to variational inequalities. Chapter IX of *Augmented Lagrangian Methods: Applications to the Solution of Boundary Value Problems* (M. Fortin and R. Glowinski, eds), North-Holland, pp. 299–340.

D. Gabay and B. Mercier (1976), 'A dual algorithm for the solution of nonlinear variational problems via finite element approximation', *Comput. Math. Appl.* **2**, 17–40.

G. Gallo, M. D. Grigoriadis and R. E. Tarjan (1989), 'A fast parametric maximum flow algorithm and applications', *SIAM J. Comput.* **18**, 30–55.

D. Geman and G. Reynolds (1992), 'Constrained restoration and the recovery of discontinuities', *IEEE Trans. PAMI* **14**, 367–383.

S. Geman and D. Geman (1984), 'Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images', *IEEE Trans. PAMI* **6**, 721–741.

P. Getreuer (2012), 'Total variation deconvolution using split Bregman', *Image Processing On Line* **2**, 158–174.

G. Gilboa, J. Darbon, S. Osher and T. Chan (2006), 'Nonlocal convex functionals for image regularization. CAM Report 06-57, UCLA.

E. Giusti (1984), *Minimal Surfaces and Functions of Bounded Variation*, Birkhäuser.

R. Glowinski and P. Le Tallec (1989), *Augmented Lagrangian and Operator-Splitting Methods in Nonlinear Mechanics*, Vol. 9 of *SIAM Studies in Applied Mathematics*, SIAM.

R. Glowinski and A. Marroco (1975), 'Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité, d'une classe de problèmes de Dirichlet non linéaires', *Rev. Française Automat. Informat. Recherche Opérationnelle Sér. Rouge Anal. Numér.* **9**(R-2), 41–76.

D. Goldfarb and S. Ma (2012), 'Fast multiple-splitting algorithms for convex optimization', *SIAM J. Optim.* **22**, 533–556.

B. Goldluecke, E. Strekalovskiy and D. Cremers (2012), 'The natural vectorial total variation which arises from geometric measure theory', *SIAM J. Imaging Sci.* **5**, 537–564.

B. Goldluecke, E. Strekalovskiy and D. Cremers (2013), 'Tight convex relaxations for vector-valued labeling', *SIAM J. Imaging Sci.* **6**, 1626–1664.

A. A. Goldstein (1964), 'Convex programming in Hilbert space', *Bull. Amer. Math. Soc.* **70**, 709–710.

T. Goldstein and S. Osher (2009), 'The split Bregman method for L^1-regularized problems', *SIAM J. Imaging Sci.* **2**, 323–343.

T. Goldstein, M. Li, X. Yuan, E. Esser and R. Baraniuk (2015), Adaptive primal–dual hybrid gradient methods for saddle-point problems. arXiv:1305.0546v2

T. Goldstein, B. O'Donoghue, S. Setzer and R. Baraniuk (2014), 'Fast alternating direction optimization methods', *SIAM J. Imaging Sci.* **7**, 1588–1623.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

M. Grasmair (2010), 'Non-convex sparse regularisation', *J. Math. Anal. Appl.* **365**, 19–28.
M. Grasmair, M. Haltmeier and O. Scherzer (2011), 'Necessary and sufficient conditions for linear convergence of ℓ^1-regularization', *Comm. Pure Appl. Math.* **64**, 161–182.
L. Grippo and M. Sciandrone (2000), 'On the convergence of the block nonlinear Gauss–Seidel method under convex constraints', *Oper. Res. Lett.* **26**, 127–136.
O. Güler (1991), 'On the convergence of the proximal point algorithm for convex minimization', *SIAM J. Control Optim.* **29**, 403–419.
O. Güler (1992), 'New proximal point algorithms for convex minimization', *SIAM J. Optim.* **2**, 649–664.
K. Guo and D. Labate (2007), 'Optimally sparse multidimensional representation using shearlets', *SIAM J. Math. Analysis* **39**, 298–318.
K. Guo, G. Kutyniok and D. Labate (2006), Sparse multidimensional representations using anisotropic dilation and shear operators. In *Wavelets and Splines: Athens 2005*, Nashboro Press, pp. 189–201.
W. Hashimoto and K. Kurata (2000), 'Properties of basis functions generated by shift invariant sparse representations of natural images', *Biological Cybernetics* **83**, 111–118.
S. Hawe, M. Kleinsteuber and K. Diepold (2013), 'Analysis operator learning and its application to image reconstruction', *IEEE Trans. Image Process.* **22**, 2138–2150.
B. He and X. Yuan (2015*a*), 'Block-wise alternating direction method of multipliers for multiple-block convex programming and beyond', *SMAI J. Comput. Math.* **1**, 145–174.
B. He and X. Yuan (2015*b*), 'On non-ergodic convergence rate of Douglas–Rachford alternating direction method of multipliers', *Numer. Math.* **130**, 567–577.
B. He and X. Yuan (2015*c*), 'On the convergence rate of Douglas–Rachford operator splitting method', *Math. Program.* A **153**, 715–722.
B. He, Y. You and X. Yuan (2014), 'On the convergence of primal–dual hybrid gradient algorithm', *SIAM J. Imaging Sci.* **7**, 2526–2537.
M. R. Hestenes (1969), 'Multiplier and gradient methods', *J. Optim. Theory Appl.* **4**, 303–320.
D. S. Hochbaum (2001), 'An efficient algorithm for image segmentation, Markov random fields and related problems', *J. Assoc. Comput. Mach.* **48**, 686–701.
T. Hohage and C. Homann (2014), A generalization of the Chambolle–Pock algorithm to Banach spaces with applications to inverse problems. arXiv:1412.0126
M. Hong, Z.-Q. Luo and M. Razaviyayn (2015), Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. In *Proc. 2015 IEEE International Conference on Acoustics, Speech and Signal Processing: ICASSP*, pp. 3836–3840.
B. K. P. Horn and B. G. Schunck (1981), 'Determining optical flow', *Artif. Intell.* **17**, 185–203.
D. Hubel and T. Wiesel (1959), 'Receptive fields of single neurones in the cat's striate cortex', *J. Physiology* **148**, 574–591.
K. Ito and K. Kunisch (1990), 'The augmented Lagrangian method for equality and inequality constraints in Hilbert spaces', *Math. Program.* **46**, 341–360.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

N. A. Johnson (2013), 'A dynamic programming algorithm for the fused Lasso and l_0-segmentation', *J. Comput. Graph. Statist* **22**, 246–260.

G. Kanizsa (1979), *Organization in Vision*, Praeger.

S. Kindermann, S. Osher and P. W. Jones (2005), 'Deblurring and denoising of images by nonlocal functionals', *Multiscale Model. Simul.* **4**, 1091–1115.

K. C. Kiwiel (1997), 'Proximal minimization methods with generalized Bregman functions', *SIAM J. Control Optim.* **35**, 1142–1168.

F. Knoll, K. Bredies, T. Pock and R. Stollberger (2011), 'Second order total generalized variation (TGV) for MRI', *Magnetic Resonance in Medicine* **65**, 480–491.

V. Kolmogorov, T. Pock and M. Rolinek (2016), 'Total variation on a tree', *SIAM J. Imaging Sci.*, accepted.

G. M. Korpelevich (1976), 'An extragradient method for finding saddle points and other problems', *Ekonom. i Mat. Metody* **12**, 747–756.

G. M. Korpelevich (1983), 'Extrapolational gradient methods and their connection with modified Lagrangians', *Ehkon. Mat. Metody* **19**, 694–703.

M. A. Krasnosel'skiĭ (1955), 'Two remarks on the method of successive approximations', *Uspekhi Mat. Nauk* (N.S.) **10**, 123–127.

A. Krizhevsky, I. Sutskever and G. E. Hinton (2012), Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: Proc. NIPS 2012*, pp. 1106–1114.

G. Kutyniok and W.-Q. Lim (2011), 'Compactly supported shearlets are optimally sparse', *J. Approx. Theory* **163**, 1564–1589.

G. Lawlor and F. Morgan (1994), 'Paired calibrations applied to soap films, immiscible fluids, and surfaces or networks minimizing other norms', *Pacific J. Math.* **166**, 55–83.

M. Lebrun, A. Buades and J. M. Morel (2013), 'A nonlocal Bayesian image denoising algorithm', *SIAM J. Imaging Sci.* **6**, 1665–1688.

Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard and L. D. Jackel (1989), Handwritten digit recognition with a backpropagation network. In *Advances in Neural Information Processing Systems 2: Proc. NIPS 1989*, pp. 396–404.

Y. LeCun, L. Bottou, Y. Bengio and P. Haffner (1998*a*), 'Gradient-based learning applied to document recognition', *Proc. IEEE* **86**, 2278–2324.

Y. LeCun, L. Bottou, G. Orr and K. Muller (1998*b*), Efficient BackProp. In *Neural Networks: Tricks of the Trade* (G. Orr and K. Muller, eds), Springer.

D. D. Lee and H. S. Seung (1999), 'Learning the parts of objects by nonnegative matrix factorization', *Nature* **401**, 788–791.

H. Lee, A. Battle, R. Raina and A. Y. Ng (2007), Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems 19* (B. Schölkopf, J. Platt and T. Hoffman, eds), MIT Press, pp. 801–808.

J. Lellmann and C. Schnörr (2011), 'Continuous multiclass labeling approaches and algorithms', *SIAM J. Imaging Sci.* **4**, 1049–1096.

J. Lellmann, F. Lenzen and C. Schnörr (2013), 'Optimality bounds for a variational relaxation of the image partitioning problem', *J. Math. Imaging Vision* **47**, 239–257.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

L. Lessard, B. Recht, and A. Packard (2016), 'Analysis and design of optimization algorithms via integral quadratic constraints', *SIAM J. Optim.* **26**, 57–95.

A. Levin, Y. Weiss, F. Durand and W. T. Freeman (2011), Efficient marginal likelihood optimization in blind deconvolution. In *Proc. 24th IEEE Conference on Computer Vision and Pattern Recognition: CVPR 2011*, pp. 2657–2664.

J. Liang, J. Fadili and G. Peyré (2014), Local linear convergence of forward–backward under partial smoothness. In *Advances in Neural Information Processing Systems 27: Proc. NIPS 2014*, pp. 1970–1978.

J. Liang, J. Fadili and G. Peyré (2015), 'Convergence rates with inexact nonexpansive operators', *Math. Program.* A. doi:10.1007/s10107-015-0964-4

Q. Lin, Z. Lu and L. Xiao (2015), An accelerated proximal coordinate gradient method and its application to regularized empirical risk minimization. *SIAM J. Optim.* **25**, 2244–2273.

P. L. Lions and B. Mercier (1979), 'Splitting algorithms for the sum of two nonlinear operators', *SIAM J. Numer. Anal.* **16**, 964–979.

B. D. Lucas and T. Kanade (1981), An iterative image registration technique with an application to stereo vision. In *Proc. 7th International Joint Conference on Artificial Intelligence: IJCAI '81*, pp. 674–679.

S. Magnússon, P. Chathuranga Weeraddana, M. G. Rabbat and C. Fischione (2014), On the convergence of alternating direction Lagrangian methods for nonconvex structured optimization problems. arXiv:1409.8033

A. Mahendran and A. Vedaldi (2015), Understanding deep image representations by inverting them. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition: CVPR 2015*, pp. 5188–5196.

J. Mairal, F. Bach, J. Ponce and G. Sapiro (2009*a*), Online dictionary learning for sparse coding. In *Proc. 26th Annual International Conference on Machine Learning: ICML '09*, ACM, pp. 689–696.

J. Mairal, J. Ponce, G. Sapiro, A. Zisserman and F. R. Bach (2009*b*), Supervised dictionary learning. In *Advances in Neural Information Processing Systems 21: Proc. NIPS 2009* (D. Koller, D. Schuurmans, Y. Bengio and L. Bottou, eds), Curran Associates, pp. 1033–1040.

S. Mallat and G. Yu (2010), 'Super-resolution with sparse mixing estimators', *IEEE Trans. Image Process.* **19**, 2889–2900.

S. Mallat and Z. Zhang (1993), 'Matching pursuits with time-frequency dictionaries', *Trans. Signal Process.* **41**, 3397–3415.

W. R. Mann (1953), 'Mean value methods in iteration', *Proc. Amer. Math. Soc.* **4**, 506–510.

B. Martinet (1970), 'Brève communication. régularisation d'inéquations variationnelles par approximations successives', *ESAIM: Mathematical Modelling and Numerical Analysis / Modélisation Mathématique et Analyse Numérique* **4**(R3), 154–158.

S. Masnou and J.-M. Morel (2006), 'On a variational theory of image amodal completion', *Rend. Sem. Mat. Univ. Padova* **116**, 211–252.

H. Mine and M. Fukushima (1981), 'A minimization method for the sum of a convex function and a continuously differentiable function', *J. Optim. Theory Appl.* **33**, 9–23.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

G. J. Minty (1962), 'Monotone (nonlinear) operators in Hilbert space', *Duke Math. J.* **29**, 341–346.

T. Möllenhoff, E. Strekalovskiy, M. Moeller and D. Cremers (2015), 'The primal–dual hybrid gradient method for semiconvex splittings', *SIAM J. Imaging Sci.* **8**, 827–857.

M. G. Mora and M. Morini (2001), 'Local calibrations for minimizers of the Mumford–Shah functional with a regular discontinuity set', *Ann. Inst. H. Poincaré Anal. Non Linéaire* **18**, 403–436.

J. L. Morales and J. Nocedal (2011), 'Remark on "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization" [mr1671706]', *ACM Trans. Math. Software* **38**, 7.

J. J. Moreau (1965), 'Proximité et dualité dans un espace Hilbertien', *Bull. Soc. Math. France* **93**, 273–299.

A. Moudafi and M. Oliny (2003), 'Convergence of a splitting inertial proximal method for monotone operators', *J. Comput. Appl. Math.* **155**, 447–454.

D. Mumford and J. Shah (1989), 'Optimal approximation by piecewise smooth functions and associated variational problems', *Comm. Pure Appl. Math.* **42**, 577–685.

S. Nam, M. Davies, M. Elad and R. Gribonval (2013), 'The cosparse analysis model and algorithms', *Appl. Comput. Harmonic Anal.* **34**, 30–56.

A. S. Nemirovski (2004), 'Prox-method with rate of convergence $O(1/t)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems', *SIAM J. Optim.* **15**, 229–251.

A. S. Nemirovski and D. Yudin (1983), *Problem Complexity and Method Efficiency in Optimization* (translated from Russian), Wiley-Interscience Series in Discrete Mathematics, Wiley.

Y. Nesterov (1983), 'A method for solving the convex programming problem with convergence rate $O(1/k^2)$', *Dokl. Akad. Nauk SSSR* **269**, 543–547.

Y. Nesterov (2004), *Introductory Lectures on Convex Optimization: A Basic Course*, Vol. 87 of *Applied Optimization*, Kluwer Academic.

Y. Nesterov (2005), 'Smooth minimization of non-smooth functions', *Math. Program.* **103**, 127–152.

Y. Nesterov (2012), 'Efficiency of coordinate descent methods on huge-scale optimization problems', *SIAM J. Optim.* **22**, 341–362.

Y. Nesterov (2013), 'Gradient methods for minimizing composite functions', *Math. Program.* B **140**, 125–161.

Y. Nesterov (2015), 'Universal gradient methods for convex optimization problems', *Math. Program.* A **152**, 381–404.

M. Nikolova (2004), 'A variational approach to remove outliers and impulse noise', *J. Math. Image Vis.* **20**, 99–120.

R. Nishihara, L. Lessard, B. Recht, A. Packard and M. I. Jordan (2015), A general analysis of the convergence of ADMM. In *Proc. 32nd International Conference on Machine Learning*, Vol. 37 of *JMLR Workshop and Conference Proceedings*.

M. Nitzberg, D. Mumford and T. Shiota (1993), *Filtering, Segmentation and Depth*, Vol. 662 of *Lecture Notes in Computer Science*, Springer.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

J. Nocedal and S. J. Wright (2006), *Numerical Optimization*, second edition, Springer Series in Operations Research and Financial Engineering, Springer.
P. Ochs, T. Brox and T. Pock (2015), 'iPiasco: Inertial proximal algorithm for strongly convex optimization', *J. Math. Imaging Vision* **53**, 171–181.
P. Ochs, Y. Chen, T. Brox and T. Pock (2014), 'iPiano: inertial proximal algorithm for nonconvex optimization', *SIAM J. Imaging Sci.* **7**, 1388–1419.
D. O'Connor and L. Vandenberghe (2014), 'Primal–dual decomposition by operator splitting and applications to image deblurring', *SIAM J. Imaging Sci.* **7**, 1724–1754.
B. O'Donoghue and E. Candès (2015), 'Adaptive restart for accelerated gradient schemes', *Found. Comput. Math.* **15**, 715–732.
B. A. Olshausen and D. J. Field (1997), 'Sparse coding with an overcomplete basis set: A strategy employed by V1?', *Vision Research* **37**, 3311–3325.
Z. Opial (1967), 'Weak convergence of the sequence of successive approximations for nonexpansive mappings', *Bull. Amer. Math. Soc.* **73**, 591–597.
Y. Ouyang, Y. Chen, G. Lan and E. Pasiliao, Jr (2015), 'An accelerated linearized alternating direction method of multipliers', *SIAM J. Imaging Sci.* **8**, 644–681.
P. Paatero and U. Tapper (1994), 'Positive matrix factorization: A nonnegative factor model with optimal utilization of error estimates of data values', *Environmetrics* **5**, 111–126.
N. Parikh and S. Boyd (2014), 'Proximal algorithms', *Found. Trends Optim.* **1**, 127–239.
G. B. Passty (1979), 'Ergodic convergence to a zero of the sum of monotone operators in Hilbert space', *J. Math. Anal. Appl.* **72**, 383–390.
P. Patrinos, L. Stella and A. Bemporad (2014), Douglas–Rachford splitting: Complexity estimates and accelerated variants. In *Proc. 2014 IEEE 53rd Annual Conference on Decision and Control: CDC*, pp. 4234–4239.
G. Peyré, S. Bougleux and L. Cohen (2008), Non-local regularization of inverse problems. In *Proc. 10th European Conference on Computer Vision: ECCV 2008*, Vol. 5304 of *Lecture Notes in Computer Science*, Springer, pp. 57–68.
G. Peyré, J. Fadili and J.-L. Starck (2010), 'Learning the morphological diversity', *SIAM J. Imaging Sci.* **3**, 646–669.
J. C. Picard and H. D. Ratliff (1975), 'Minimum cuts and related problems', *Networks* **5**, 357–370.
T. Pock and A. Chambolle (2011), Diagonal preconditioning for first order primal–dual algorithms. In *Proc. IEEE International Conference on Computer Vision: ICCV 2011*, IEEE, pp. 1762–1769.
T. Pock and S. Sabach (2016), Inertial proximal alternating linearized minimization (iPALM) for nonconvex and nonsmooth problems. Technical report.
T. Pock, D. Cremers, H. Bischof and A. Chambolle (2009), An algorithm for minimizing the Mumford–Shah functional. In *Proc. IEEE 12th International Conference on Computer Vision: ICCV 2009*, IEEE, pp. 1133–1140.
T. Pock, D. Cremers, H. Bischof and A. Chambolle (2010), 'Global solutions of variational models with convex regularization', *SIAM J. Imaging Sci.* **3**, 1122–1145.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

T. Pock, T. Schoenemann, G. Graber, H. Bischof and D. Cremers (2008), A convex formulation of continuous multi-label problems. In *Proc. 10th European Conference on Computer Vision: ECCV 2008*, Vol. 5304 of *Lecture Notes in Computer Science*, Springer, pp. 792–805.

B. T. Polyak (1987), *Introduction to Optimization* (translated from Russian), Translations Series in Mathematics and Engineering, Optimization Software.

L. D. Popov (1981), A modification of the Arrow–Hurwitz method of search for saddle points with an adaptive procedure for determining the iteration step. In *Classification and Optimization in Control Problems*, Akad. Nauk SSSR Ural. Nauchn. Tsentr, Sverdlovsk, pp. 52–56.

M. J. D. Powell (1969), A method for nonlinear constraints in minimization problems. In *Optimization: Keele 1968*, Academic, pp. 283–298.

M. Protter, I. Yavneh and M. Elad (2010), 'Closed-form MMSE estimation for signal denoising under sparse representation modelling over a unitary dictionary'. *IEEE Trans. Signal Process.* **58**, 3471–3484.

N. Pustelnik, C. Chaux and J.-C. Pesquet (2011), 'Parallel proximal algorithm for image restoration using hybrid regularization', *IEEE Trans. Image Process.* **20**, 2450–2462.

H. Raguet, J. Fadili and G. Peyré (2013), 'A generalized forward–backward splitting', *SIAM J. Imaging Sci.* **6**, 1199–1226.

R. T. Rockafellar (1976), 'Monotone operators and the proximal point algorithm', *SIAM J. Control Optim.* **14**, 877–898.

R. T. Rockafellar (1997), *Convex Analysis*, Princeton Landmarks in Mathematics, Princeton University Press.

C. Rother, V. Kolmogorov and A. Blake (2004), '"GrabCut": Interactive foreground extraction using iterated graph cuts', *ACM Trans. Graph.* **23**, 309–314.

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

S. Salzo and S. Villa (2012), 'Inexact and accelerated proximal point algorithms', *J. Convex Anal.* **19**, 1167–1192.

G. Sapiro and D. L. Ringach (1996), 'Anisotropic diffusion of multivalued images with applications to color filtering', *IEEE Trans. Image Process.* **5**, 1582–1586.

H. Schaefer (1957), 'Über die Methode sukzessiver Approximationen', *Jber. Deutsch. Math. Verein.* **59**, 131–140.

M. Schmidt, N. L. Roux and F. R. Bach (2011), Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in Neural Information Processing Systems 24: Proc. NIPS 2011* (J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira and K. Weinberger, eds), Curran Associates, pp. 1458–1466.

T. Schoenemann and D. Cremers (2007), Globally optimal image segmentation with an elastic shape prior. In *Proc. 11th IEEE International Conference on Computer Vision: ICCV 2007*, IEEE, pp. 1–6.

T. Schoenemann, S. Masnou and D. Cremers (2011), 'The elastic ratio: Introducing curvature into ratio-based image segmentation', *IEEE Trans. Image Process.* **20**, 2565–2581.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

S. Setzer (2011), 'Operator splittings, Bregman methods and frame shrinkage in image processing', *Internat. J. Comput. Vis.* **92**, 265–280.
R. Shefi and M. Teboulle (2014), 'Rate of convergence analysis of decomposition methods based on the proximal method of multipliers for convex minimization', *SIAM J. Optim.* **24**, 269–297.
E. Y. Sidky, C.-M. Kao and X. Pan (2006), 'Accurate image reconstruction from few-views and limited-angle data in divergent-beam CT', **14**, 119–139.
K. Simonyan and A. Zisserman (2015), Very deep convolutional networks for large-scale image recognition. In *Proc. International Conference on Learning Representations*. arXiv:1409.1556
J.-L. Starck, F. Murtagh and J. M. Fadili (2010), *Sparse Image and Signal Processing: Wavelets, Curvelets, Morphological Diversity*, Cambridge University Press.
G. Steidl and T. Teuber (2010), 'Removing multiplicative noise by Douglas–Rachford splitting methods', *J. Math. Imaging Vision* **36**, 168–184.
L. Stella, A. Themelis and P. Patrinos (2016), Forward–backward quasi-Newton methods for nonsmooth optimization problems. arXiv:1604.08096v1
G. Strang (1983), 'Maximal flow through a domain', *Math. Program.* **26**, 123–143.
G. Strang (2010), 'Maximum flows and minimum cuts in the plane', *J. Global Optim.* **47**, 527–535.
E. Strekalovskiy, A. Chambolle and D. Cremers (2014), 'Convex relaxation of vectorial problems with coupled regularization', *SIAM J. Imaging Sci.* **7**, 294–336.
P. Tan (2016), Acceleration of saddle-point methods in smooth cases. In preparation.
S. Tao, D. Boley and S. Zhang (2015), Local linear convergence of ISTA and FISTA on the LASSO problem. arXiv:1501.02888
M. Teboulle (1992), 'Entropic proximal mappings with applications to nonlinear programming', *Math. Oper. Res.* **17**, 670–690.
R. Tibshirani (1996), 'Regression shrinkage and selection via the lasso', *J. Royal Statist. Soc.* B **58**, 267–288.
P. Tseng (1991), 'Applications of a splitting algorithm to decomposition in convex programming and variational inequalities', *SIAM J. Control Optim.* **29**, 119–138.
P. Tseng (2000), 'A modified forward–backward splitting method for maximal monotone mappings', *SIAM J. Control Optim.* **38**, 431–446.
P. Tseng (2001), 'Convergence of a block coordinate descent method for nondifferentiable minimization', *J. Optim. Theory Appl.* **109**, 475–494.
P. Tseng (2008), On accelerated proximal gradient methods for convex–concave optimization. http://www.csie.ntu.edu.tw/~b97058/tseng/papers/apgm.pdf
P. Tseng and S. Yun (2009), 'Block-coordinate gradient descent method for linearly constrained nonsmooth separable optimization', *J. Optim. Theory Appl.* **140**, 513–535.
T. Valkonen (2014), 'A primal–dual hybrid gradient method for nonlinear operators with applications to MRI', *Inverse Problems* **30**, 055012.
T. Valkonen and T. Pock (2015), Acceleration of the PDHGM on strongly convex subspaces. arXiv:1511.06566

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

V. N. Vapnik (2000), *The Nature of Statistical Learning Theory*, second edition, Statistics for Engineering and Information Science, Springer.

R. S. Varga (1962), *Matrix Iterative Analysis*, Prentice Hall.

S. Villa, S. Salzo, L. Baldassarre and A. Verri (2013), 'Accelerated and inexact forward–backward algorithms', *SIAM J. Optim.* **23**, 1607–1633.

B. C. Vũ (2013*a*), 'A splitting algorithm for dual monotone inclusions involving cocoercive operators', *Adv. Comput. Math.* **38**, 667–681.

B. C. Vũ (2013*b*), 'A variable metric extension of the forward–backward–forward algorithm for monotone operators', *Numer. Funct. Anal. Optim.* **34**, 1050–1065.

Y. Wang, W. Yin and J. Zeng (2015), Global convergence of ADMM in nonconvex nonsmooth optimization. arXiv:1511.06324

M. Yamagishi and I. Yamada (2011), 'Over-relaxation of the fast iterative shrinkage-thresholding algorithm with variable stepsize', *Inverse Problems* **27**, 105008.

F. Yanez and F. Bach (2014), Primal–dual algorithms for non-negative matrix factorization with the Kullback–Leibler divergence. arXiv:1412.1788

W. Yin and S. Osher (2013), 'Error forgetting of Bregman iteration', *J. Sci. Comput.* **54**, 684–695.

G. Yu, G. Sapiro and S. Mallat (2012), 'Solving inverse problems with piecewise linear estimators: from Gaussian mixture models to structured sparsity', *IEEE Trans. Image Process.* **21**, 2481–2499.

R. Zabih and J. Woodfill (1994), Non-parametric local transforms for computing visual correspondence. In *Proc. 3rd European Conference on Computer Vision: ECCV '94*, Vol. II, Vol. 801 of *Lecture Notes in Computer Science*, Springer, pp. 151–158.

C. Zach, D. Gallup, J. M. Frahm and M. Niethammer (2008), Fast global labeling for real-time stereo using multiple plane sweeps. In *Vision, Modeling, and Visualization 2008* (O. Deussen, D. Keim and D. Saupe, eds), IOS Press, pp. 243–252.

C. Zach, T. Pock and H. Bischof (2007), A duality based approach for realtime TV-L^1 optical flow. In *Proc. 29th DAGM Symposium on Pattern Recognition*, Vol. 4713 of *Lecture Notes in Computer Science*, Springer, pp. 214–223.

S. K. Zavriev and F. V. Kostyuk (1991), The heavy ball method in nonconvex optimization problems. In *Software and Models of Systems Analysis* (in Russian), Moskov. Gos. Univ., Moscow, pp. 179–186, 195. Translation in *Comput. Math. Model.* **4** (1993), 336–341.

M. Zeiler, D. Krishnan, G. Taylor and R. Fergus (2010), Deconvolutional networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition: CVPR 2010*, pp. 2528–2535.

X. Zhang, M. Burger and S. Osher (2011), 'A unified primal–dual algorithm framework based on Bregman iteration', *J. Sci. Comput.* **46**, 20–46.

X. Zhang, M. Burger, X. Bresson and S. Osher (2010), 'Bregmanized nonlocal regularization for deconvolution and sparse reconstruction', *SIAM J. Imaging Sci.* **3**, 253–276.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press

C. Zhu, R. H. Byrd, P. Lu and J. Nocedal (1997), 'Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization', *ACM Trans. Math. Software* **23**, 550–560.

M. Zhu and T. Chan (2008), An efficient primal–dual hybrid gradient algorithm for total variation image restoration. CAM Report 08-34, UCLA.

https://doi.org/10.1017/S096249291600009X Published online by Cambridge University Press