Title
Primal-dual decomposition by operator splitting and applications to image deblurring

Author
O'Connor, Daniel Verity

Publication Date
2015

Peer reviewed|Thesis/dissertation

University of California
Los Angeles

Primal-dual decomposition by operator splitting and applications to image deblurring

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Mathematics

by

Daniel Verity O'Connor

2015

Abstract of the Dissertation

Primal-dual decomposition by operator splitting and applications to image deblurring

by

Daniel Verity O'Connor

Doctor of Philosophy in Mathematics

University of California, Los Angeles, 2015

Professor Lieven Vandenberghe, Chair

We present primal-dual decomposition algorithms for convex optimization problems with cost functions $f(x)+g(Ax)$, where f and g have inexpensive proximal operators and A can be decomposed as a sum of structured matrices. The methods are based on the Douglas-Rachford splitting algorithm applied to various splittings of the primal-dual optimality conditions. We discuss applications to image deblurring problems with non-quadratic data fidelity terms, different types of convex regularization, and simple convex constraints. In these applications, the primal-dual splitting approach allows us to handle general boundary conditions for the blurring operator. We also give a domain decomposition approach to image deblurring, where the blur operator may use a different blur kernel for each image patch, and at each iteration all patches are processed in parallel. Numerical results indicate that the primal-dual splitting methods compare favorably with the alternating direction method of multipliers, the Douglas-Rachford algorithm applied to a reformulated primal problem, and the Chambolle-Pock primal-dual algorithm.

A key property exploited in most image deblurring methods is spatial invariance of the blurring operator, which makes it possible to use the fast Fourier transform (FFT) when solving linear equations involving the operator. In this thesis we extend this approach to two popular models for space-varying blurring operators, the Nagy-O'Leary model and the Efficient Filter Flow model. We show how splitting methods derived from the Douglas-

Rachford algorithm can be implemented with a low complexity per iteration, dominated by a small number of FFTs.

The dissertation of Daniel Verity O'Connor is approved.

Christopher R. Anderson

Stanley J. Osher

Yingnian Wu

Lieven Vandenberghe, Committee Chair

University of California, Los Angeles

2015

To Seririthanar

Table of Contents

List of Figures

List of Tables

Acknowledgments

I'd like to thank first of all Lieven Vandenberghe, who provided outstanding research guidance and also truly cares about mentoring his students. I couldn't have asked for a better PhD advisor.

I'd also like to thank my committee members, Chris Anderson, Stan Osher, and Ying Nian Wu, who have been extremely supportive and who have played a key role in making my PhD possible.

I am very fortunate to have the opportunity to be a part of the UCLA math community, where everyone has always been welcoming and accessible. Wotao Yin, Olga Radko, Stefano Soatto, Zhuowen Tu, Alan Yuille, Luminita Vese, and Andrea Bertozzi were all immensely helpful throughout my PhD. I'd especially like to thank Maggie Albert who made all the difference during my time at UCLA. I'm very grateful for collaborations with Dan Nguyen, Ke Sheng, Nick Dwork, and Genna Smith.

I'd like to thank Eric Shrader for his invaluable mentorship at Arete Associates.

Finally I'd like to thank my parents for their heroic levels of support and encouragement.

Vita

2001	B.S. (Mathematics), University of Texas at Austin
2003	M.A. (Mathematics), University of California, Los Angeles
2004-2005	Actuarial Trainee, Health Net, Inc.
2006–2009	Research Analyst, Arete Associates.
2009-2015	Graduate Student Researcher, University of California, Los Angeles

Publications

Journal papers

- D. O'Connor and L. Vandenberghe, *Primal-dual decomposition by operator splitting and applications to image deblurring*, SIAM J. Imaging Sci., 7(3),1724-1754,2014.
- D. Nguyen, D. O'Connor, V. Yu, D. Ruan, M. Cao, D. Low, K. Sheng, *Dose domain regularization of MLC leaf patterns for highly complex IMRT plans*, Medical Physics, 42(4),2015.
- D. O'Connor and L. Vandenberghe, *Total variation image deblurring with space-varying kernel.* To be submitted.

Conference abstracts and presentations

- D. O'Connor, L. Vandenberghe, *Total variation image deblurring with space-varying kernel via Douglas-Rachford splitting.* Presentation to be given at ISMP 2015.
- N. Dwork, D. O'Connor, N. Addy, R. Ingle, J. Pauly, D. Nishimura, *Using optical flow to estimate displacement between 3D navigators in coronary angiography.* Abstract accepted to ISMRM 2015.
- D. Nguyen, D. O'Connor, V. Yu, D. Ruan, M. Cao, D. Low, K. Sheng, *A new intensity modulation radiation therapy (IMRT) optimizer solution with robust fluence maps for MLC segmentation.* Abstract accepted as a talk at AAPM 2015.
- D. Nguyen, D. Ruan, D. O'Connor, D. Low, S. Boucher, K. Sheng, *A novel Haar wavelet based approach to deliver non-coplanar intensity modulated radiotherapy using sparse orthogonal collimators.* Abstract accepted as a talk at AAPM 2015.
- D. Nguyen, V. Yu, D. Ruan, H. Semwal, D. O'Connor, M. Cao, D. Low, K. Sheng, *Dose domain optimization of MLC Leaf Patterns for highly complicated* 4π *IMRT plans.* Abstract accepted as a talk at AAPM 2014.
- N. Dwork, G. Smith, D. O'Connor, U. Sikora, K. Lurie, J. Pauly, A. Ellerbee, *Extraction of the Attenuation Coefficient from OCT Data.* Poster presented at the 2014 National Training Meeting of the NIBIB.

CHAPTER 1

Introduction

1.1 Composite convex optimization

In this thesis we focus on solving convex optimization problems of the canonical form

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad f(x) + g(Ax) \tag{1.1}$$

where f and g are closed (*i.e.*, lower semicontinuous) convex functions and A is a matrix. This problem form (sometimes known as a "composite convex optimization problem") has been widely studied in the convex optimization literature [Roc74, Gab83, CP11a]; for example, the Fenchel-Rockafellar approach to duality is based on this problem form [Roc67]. Problem (1.1) is simple enough that its analysis is clean and elegant, and yet expressive enough that many important practical problems arising in application areas such as image processing, medical imaging, machine learning, and signal processing can be expressed in this form with *simple* choices of f, g and structured matrices A. By "simple", we mean that f and g have "proximal operators" [Mor65] that can be evaluated efficiently. The proximal operator (with parameter $t > 0$) of a proper closed convex function f is denoted by prox_{tf}, and is defined by

$$\text{prox}_{tf}(x) = \underset{u}{\text{argmin}}\ f(u) + \frac{1}{2t}\|u - x\|_2^2. \tag{1.2}$$

For example, if f is the indicator function of a closed convex set Ω, so that

$$f(x) = \begin{cases} 0 & \text{if } x \in \Omega \\ \infty & \text{otherwise,} \end{cases}$$

then $\text{prox}_{tf}(x) = P_\Omega(x)$, the projection of x onto Ω. This projection can be computed inexpensively in many important cases. Another important example is the case where $f(x) = \|x\|$ for some norm $\|\cdot\|$. In this case,

$$\text{prox}_{tf}(x) = x - P_{tC}(x),$$

where C is the dual norm unit ball. Proximal operators will be discussed in more detail in section 2.3. The algorithms presented in this thesis are "proximal algorithms", in that they are based on or make use of proximal operators.

One basic type of structure in A that we will consider is simply that the matrix-vector products Ax and $A^T z$ can be computed efficiently, for any vectors x and z. This is the structure exploited by forward-backward methods [Tse00, CW05] and semi-implicit primal-dual methods [EZC09, Ess10, BC11b, HY12, Con13], of which the Chambolle-Pock algorithm is the best known example. Any sparse matrix has this structure, for example. The type of structure that we will focus on mainly in this thesis is that of a matrix A which is a sum of matrices A_i such that linear systems with coefficient $I + tA_i^T A_i$ can be solved efficiently. Linear systems of this form are encountered frequently in proximal algorithms, and this type of structured matrix appears in many applications. For example, if A_i is a discrete Fourier transform matrix, then linear systems with coefficient $I + tA_i^T A_i$ can be solved efficiently via the fast Fourier transform.

Here we list some examples to show the expressive power of problem (1.1).

Linear equality constraints The problem

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & Ax = b \end{aligned}$$

can be expressed in the form (1.1) if we choose g to be the the indicator function of the set $\{b\}$.

Convex set constraints Problems with convex set constraints, such as

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & Ax - b \in C \end{aligned}$$

where C is a convex set, have the form (1.1) where g is the indicator function of C. By taking f to be linear and taking C to be a Cartesian product of convex cones, we see that cone programming problems including linear programs, second-order cone programs and semidefinite programs can be expressed in the form (1.1). By taking C to be the indicator function for a norm ball, we can express constraints of the form $\|Ax - b\| \leq \epsilon$.

Norm regularization Regularization terms that penalize a norm of x can be expressed via the function f. If $f(x) = \|x\|_1$, then problem (1.1) becomes

$$\underset{x}{\text{minimize}} \quad g(Ax) + \|x\|_1.$$

An important special case is where $g(y) = (\rho/2)\|y - b\|_2^2$, in which case (1.1) is the LASSO problem:

$$\underset{x}{\text{minimize}} \quad \frac{\rho}{2}\|Ax - b\|_2^2 + \|x\|_1.$$

Another important special case is where g is the indicator function of $\{b\}$. In this case, we get the basis pursuit problem [CDS01]

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \|x\|_1 \\ \text{subject to} \quad & Ax = b. \end{aligned}$$

In the next section, we will discuss image deblurring problems and show how they can be expressed in the canonical form (1.1) with simple functions f, g and structured A. We will see that the canonical problem form (1.1) is particularly well-suited for image deblurring problems.

1.2 Image deblurring by convex optimization

We first discuss the blurring model and express the deblurring problem in a general optimization problem of the form (1.1). Let b be a vector containing the pixel intensities of an $N \times N$ blurry, noisy image, stored in column-major order as a vector of length $n = N^2$. Assume b is generated by a linear blurring operation with additive noise, *i.e.*,

$$b = Kx_{\mathrm{t}} + w, \tag{1.3}$$

where K is the blurring operator, $x_{\mathrm{t}} \in \mathbb{R}^n$ is the unknown true image, and w is noise. The deblurring problem is to estimate x_{t} from b. Since blurring operators are often very ill-conditioned the solution $x = K^{-1}b$ is a poor estimate of x_{t} and regularization or constraints must be applied to obtain better estimates [HNO06]. We will formulate the deblurring problem as an optimization problem of the following general form:

$$\text{minimize} \quad \phi_{\mathrm{f}}(Kx - b) + \phi_{\mathrm{r}}(x) + \phi_{\mathrm{s}}(Dx), \tag{1.4}$$

with ϕ_{f}, ϕ_{s}, and ϕ_{r} convex penalty or indicator functions. This optimization problem is a special case of (1.1) with

$$f(x) = \phi_{\mathrm{r}}(x), \qquad g(u, v) = \phi_{\mathrm{f}}(u - b) + \phi_{\mathrm{s}}(v), \qquad A = \begin{bmatrix} K \\ D \end{bmatrix}.$$

We now discuss the three terms in (1.4) in more detail.

The first term in (1.4) is called the data fidelity term, and penalizes or limits the deviation between the observed image and the model output for the reconstructed image. Typical choices for ϕ_{f} include a quadratic penalty $\phi_{\mathrm{f}}(u) = (1/2)\|u\|^2$ with $\|\cdot\|$ the Euclidean norm, the 1-norm penalty $\phi_{\mathrm{f}}(u) = \|u\|_1$, or the Huber penalty

$$\phi_{\mathrm{f}}(u) = \sum_{k=1}^{n} h_\eta(u_k), \qquad h_\eta(v) = \begin{cases} v^2/(2\eta) & |v| \le \eta \\ |v| - \eta/2 & |v| \ge \eta. \end{cases}$$

One can also consider an indicator function $\phi_{\mathrm{f}}(u) = \delta_S(u)$ of a closed convex set S. For example, if S is a Euclidean norm ball $S = \{u \mid \|u\| \le \sigma\}$ and we take $\phi_{\mathrm{f}} = \delta_S$, the problem

is equivalent to

$$\begin{array}{ll} \text{minimize} & \phi_{\mathrm{s}}(Dx) + \phi_{\mathrm{r}}(x) \\ \text{subject to} & \|Kx - b\| \leq \sigma. \end{array}$$

More generally, ϕ_{f} can be a penalty function with a non-trivial domain, such as $\phi_{\mathrm{f}}(u) = -\sum_i \log(1 - u_i^2)$ with $\mathbf{dom}\, \phi_{\mathrm{f}} = \{u \mid \|u\|_\infty < 1\}$.

Practical algorithms must exploit the fact that the blurring matrix K is highly structured. For example, if K represents convolution with a spatially invariant point spread function and periodic boundary conditions are used, then it is block-circulant with circulant blocks. It therefore has a spectral decomposition $K = Q^H \mathbf{diag}(\lambda) Q$, where $Q = W \otimes W$ is the Kronecker product of the length-N DFT matrix with itself, and λ is the 2-dimensional DFT of the convolution kernel [HNO06]. If the point spread function is also doubly symmetric and reflexive boundary conditions are used, then K can be diagonalized in a similar way by multiplication with orthogonal two-dimensional discrete cosine transform matrices [HNO06, NCT99]. For other types of boundary conditions (for example, zero or replicate boundary conditions) K can be expressed as a sum $K = K_{\mathrm{c}} + K_{\mathrm{s}}$ where K_{c} is the blurring operator for periodic or reflexive boundary conditions, and K_{s} is a sparse term that corrects the values near the boundary.

The second term in (1.4) represents a regularization term or a constraint on x. Typical choices for ϕ_{r} are a quadratic penalty $\phi_{\mathrm{r}}(x) = \gamma\|x\|^2/2$, or the indicator function of a convex set (for example, box constraints $x \in [0, 1]^n$, added to limit the range of the pixel intensities). As pointed out in [Vog02, BT09, CTY13] the explicit addition of box or nonnegativity constraints on x can improve the quality of the restoration substantially.

The last term in (1.4) can be used to add a smoothing penalty. In that case the matrix $D \in \mathbb{R}^{2n \times n}$ is defined as concatenated vertical and horizontal discretized derivative operators.

With periodic boundary conditions, D is given by

$$D = \begin{bmatrix} I \otimes D_1 \\ D_1 \otimes I \end{bmatrix}, \qquad D_1 = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ 0 & 0 & -1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -1 & 1 \\ 1 & 0 & 0 & \cdots & 0 & -1 \end{bmatrix} \in \mathbb{R}^{n \times n}.$$

If we choose for $\phi_{\mathrm{s}} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ the norm

$$\phi_{\mathrm{s}}(u, v) = \gamma \|(u, v)\|_{\mathrm{iso}} = \gamma \sum_{k=1}^{n} \sqrt{u_k^2 + v_k^2}$$

then $\phi_{\mathrm{s}}(Dx) = \gamma\,\mathrm{TV}(x)$ is the total variation penalty introduced by Rudin, Osher, and Fatemi [ROF92]. For quadratic ϕ_{f} and zero ϕ_{r} the general problem (1.4) then reduces to the classic TV-regularized deblurring problem

$$\text{minimize} \quad \frac{1}{2}\|Kx - b\|^2 + \gamma\,\mathrm{TV}(x). \tag{1.5}$$

Another interesting choice for the third term in (1.4) is $\phi_{\mathrm{s}}(Dx) = \|Dx\|_1$ where D represents a wavelet or shearlet transform matrix.

In summary, the problem format (1.4) includes a variety of useful formulations of the image deblurring problem. In all these examples the functions ϕ_{f}, ϕ_{r}, and ϕ_{s} are convex and relatively simple (*i.e.*, have inexpensive prox-operators), but they are not necessarily quadratic or differentiable, and they can have a restricted domain. Another challenging aspect of these image deblurring problems is that they are typically very large scale.

1.3 Proximal algorithms

Beginning with the discovery of Karmarkar's algorithm in 1984 [Kar84], a class of algorithms known as interior point methods [NN94, Wri97, BN01] was studied intensely by convex optimization researchers for a period of about 20 years (and much research still goes on in

this area). ("A lot was predicted for that year [1984], but I doubt that Orwell expected much of a revolution in linear programming." – Gilbert Strang [Str87]) Interior point methods are currently in a very mature state – they are applicable to a wide range of problems, can handle problems involving thousands of variables (or more, depending on problem structure), have nice theoretical convergence guarantees, and perform very well in practice. Additionally, high quality software implementations of interior point methods are available.

However, interior point methods have the disadvantage that they require solving a large linear system of equations at each iteration. For very large scale problems, involving hundreds of thousands or millions of variables, solving these linear systems at each iteration becomes prohibitively expensive. Hence, in the past several years, much research activity in convex optimization has focused on developing "first-order methods," including so-called "proximal algorithms" [PB13, CP10, CP11b], for solving very large scale, constrained, nondifferentiable convex optimization problems arising in areas such as image processing, medical imaging, compressed sensing, machine learning, and control theory. A prototypical first-order method is gradient descent, which notably does not require solving a linear system at each iteration (unlike another classical method, Newton's method).

Proximal algorithms make use of proximal operators, defined in equation (1.2). The most basic proximal algorithm is the "proximal point method" [Roc76b], which minimizes a (lower semicontinuous) convex function f via the iteration

$$x^{k+1} = \text{prox}_{tf}(x^k).$$

This method is useful in situations where evaluating the proximal operator of f is significantly less expensive than minimizing f. While such situations are very uncommon, the proximal point method can be adapted to give a practical algorithm by allowing inexact proximal operator evaluations. The augmented Lagrangian method can be interpreted as solving the dual problem via the proximal point method. The proximal point method also serves as a prototype for other more sophisticated and more useful proximal algorithms.

An extremely useful proximal algorithm is the Douglas-Rachford method [LM79, Gab83,

EB92, CP07, BC11a, BPC11, PB13], which minimizes $f(x) + g(x)$ (where f, g are proper closed convex functions) via the iteration

$$x^{k+1} = \text{prox}_{tf}(z^k) \tag{1.6}$$

$$y^{k+1} = \text{prox}_{tg}(2x^{k+1} - z^k) \tag{1.7}$$

$$z^{k+1} = z^k + \rho(y^{k+1} - x^{k+1}). \tag{1.8}$$

Under mild assumptions, the variable x^k is guaranteed to converge to a minimizer of $f + g$ (assuming a minimizer exists), for any choice of the parameters $t > 0$ and $\rho \in (0, 2)$ [EB92]. Notice that the Douglas-Rachford iteration requires us to evaluate the proximal operators of f and g separately, but we are never required to evaluate the proximal operator of $f + g$. Thus, the Douglas-Rachford method is useful for minimizing a function that is a sum of two "simple" functions.

The Douglas-Rachford method is closely related to the Alternating Direction Method of Multipliers (ADMM) / Split Bregman method [GM75, GM76, GO09, BPC11, Eck12]. ADMM solves the problem

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & f(x) + g(y) \\ \text{subject to} \quad & Ax + By = c \end{aligned} \tag{1.9}$$

via the iteration

$$\begin{aligned} x^{k+1} &= \underset{x}{\text{argmin}}\ L(x, y^k, z^k) \\ y^{k+1} &= \underset{y}{\text{argmin}}\ L(x^{k+1}, y, z^k) \\ z^{k+1} &= z^k + t(Ax^{k+1} + By^{k+1} - c), \end{aligned}$$

where L is the "augmented Lagrangian"

$$L(x, y, z) = f(x) + g(y) + \langle Ax + By - c, z\rangle + \frac{t}{2}\|Ax + By - c\|_2^2.$$

Under mild assumptions, the iterates x^k and y^k are guaranteed to converge to optimal values of x and y (assuming optimal values exist) for any choice of the parameter $t > 0$. As shown in

[Gab83, EB92], ADMM / Split Bregman can be interpreted as using the Douglas-Rachford method to solve the dual of problem (5.9). Conversely, the Douglas-Rachford method is equivalent to minimizing $f + g$ by applying ADMM to the reformulated problem

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & f(x) + g(y) \\ \text{subject to} \quad & x - y = 0. \end{aligned}$$

As we will see in chapter 2, the iteration (1.6) is in fact a special case of a more general Douglas-Rachford method that is able to solve monotone inclusion problems (a class of problems that includes convex optimization problems). Most of the algorithms presented in this thesis are based on the Douglas-Rachford method.

1.4 Decomposition in convex optimization

In this section we explain how algorithms for solving problems of the canonical form (1.1), in the case where A is a sum of structured matrices, can be viewed as extensions of the classical ideas of primal and dual decomposition [Las70, BT89]. Classically, the terms "primal decomposition", "dual decomposition", and "primal-dual decomposition" refer to special techniques for solving "almost separable" convex optimization problems – that is, problems which would separate nicely into independent subproblems if not for the presence of some coupling variables, some coupling constraints, or a combination of both coupling variables and coupling constraints. For example, in the problem

$$\begin{aligned} \underset{x_1,\ldots,x_m,x_{m+1}}{\text{minimize}} \quad & \sum_{i=1}^{m+1} f_i(x_i) \\ \text{subject to} \quad & A_{ii}x_i + A_{i,m+1}x_{m+1} = b_i, \qquad i = 1,\ldots,m \\ & \sum_{i=1}^{m+1} A_{m+1,i}x_i = b_{m+1}, \end{aligned} \tag{1.10}$$

the variable x_{m+1} is a coupling variable, and the constraint $\sum_{i=1}^{m+1} A_{m+1,i}x_i = b_{m+1}$ is a coupling constraint.

Problem (1.10) can be expressed in the form (1.1), if we take

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \\ x_{m+1} \end{bmatrix}, \quad A = \begin{bmatrix} A_{11} & & & & A_{1,m+1} \\ & A_{22} & & & A_{2,m+1} \\ & & \ddots & & \vdots \\ & & & A_{m,m} & A_{m,m+1} \\ A_{m+1,1} & A_{m+1,2} & \cdots & A_{m+1,m} & A_{m+1,m+1} \end{bmatrix}, \tag{1.11}$$

and

$$f(x) = \sum_{i=1}^{m+1} f_i(x_i), \qquad g(y) = \sum_{i=1}^{m+1} \delta_{\{b_i\}}(y_i),$$

where $\delta_{\{b_i\}}$ is the indicator function for the set $\{b_i\}$ and y is the concatenation of vectors $y_1, \ldots, y_{m+1}$. Expressing problem (1.10) in the canonical form (1.1) suggests some ways to generalize the classical notion of primal-dual decomposition. The matrix A can be decomposed as $A = B + C$, where

$$B = \begin{bmatrix} A_{11} & & & & \\ & A_{22} & & & \\ & & \ddots & & \\ & & & A_{m,m} & \\ & & & & A_{m+1,m+1} \end{bmatrix}, \; C = \begin{bmatrix} & & & & A_{1,m+1} \\ & & & & A_{2,m+1} \\ & & & & \vdots \\ & & & & A_{m,m+1} \\ A_{m+1,1} & A_{m+1,2} & \cdots & A_{m+1,m} & \end{bmatrix}.$$

The essential structure here (in our approach, at least) is that the matrix B is block diagonal, the matrix C is sparse, and the functions f and g are block separable (conformably with B). A first generalization is to allow C to be any sparse matrix, so that $B + C$ is "block diagonal + sparse" but does not necessarily have the arrow structure seen in equation (1.11). We can also allow the functions $g_i(y_i)$ to be any "simple" functions, not necessarily indicator functions. A further generalization is to consider problems where B and C each have some structure that can be exploited, though not necessarily block diagonal or sparse structure, and f and g are "simple" but not necessarily block separable. Algorithms for solving convex optimization problems with this structure will be presented in chapter 3.

1.5 Contributions

In chapter 3, we present methods based on the Douglas-Rachford splitting algorithm for handling additive structure in A when solving problem (1.1). The key idea is to express the KKT conditions as a particular monotone inclusion problem, and note that the relevant monotone operator splits nicely. We apply these methods to the problem of image deblurring by decomposing a blur operator K as $K = K_c + K_s$, where K_c is a convolution operator (using periodic boundary conditions) and K_s is a sparse operator that only affects pixel values near the image boundary. We also show how a simple type of spatially variant blur operator can be handled in this framework by decomposing the blur operator as a sum of a block diagonal term and a sparse term.

When applied to problems with the structure (1.10), the methods presented in chapter 3 overcome many of the difficulties of the classical approach to primal-dual decomposition, namely, that the dual function might be nondifferentiable and might have a nontrivial domain (so that subproblems we are required to solve might be unbounded below), and also that the projected subgradient method converges very slowly.

In chapter 4 we consider the problem of non-blind image deblurring with a spatially variant model of blur. We show that for two fundamental models of spatially variant blur – the Nagy-O'Leary model and the related Efficient Filter Flow model – it is possible to express the image deblurring in the form (1.1), in such a way that the proximal operators of f and g can be evaluated efficiently. For both blur models, the key point will be the evaluation of the proximal operator of g. This approach yields algorithms that, despite the use of spatially variant blur models, are able to make use of the Fast Fourier Transform to solve total variation regularized image deblurring problems with an efficiency comparable to that of algorithms that assume a spatially invariant model of blur.

In chapter 5 we present a different method for handling additive structure in A, based on using consensus Douglas-Rachford methods for monotone inclusion problems to solve the primal-dual optimality conditions. This approach has the advantage that it naturally

handles the case where A is a sum of N structured matrices. We show an experiment where this approach compares favorably with the methods of chapter 3, and we derive a domain decomposition approach to non-blind image deblurring that uses consensus Douglas-Rachford to solve the primal-dual optimality conditions, which in this case involve a sum of 5 monotone operators. We also derive consensus versions of ADMM, by solving the dual problem with consensus Douglas-Rachford methods.

CHAPTER 2

Convex Optimization Background

2.1 Elements of convex analysis

2.1.1 The subdifferential

Let V be a finite-dimensional inner product space over $\mathbb{R}$ and let $f : V \to \mathbb{R} \cup \{\infty\}$. To say that $g \in V$ is a subgradient of f at a point $x \in V$ means that

$$f(y) \geq f(x) + \langle g, y - x \rangle \quad \text{for all } y \in V. \tag{2.1}$$

The set of all subgradients of f at x is denoted $\partial f(x)$. Subgradients can be viewed as a substitute for gradients (when working with nondifferentiable functions), and inequality (2.1) can be viewed as a substitute for the approximation $f(y) \approx f(x) + \langle \nabla f(x), y - x \rangle$ (when y is near x) which is fundamental for studying differentiable functions. Although the definition of a subgradient does not require f to be convex, subgradients are most useful when studying convex functions. This is partly due to the fact that a convex function is guaranteed to have a subgradient at any point in the relative interior of its effective domain.

The subdifferential ∂f is a set-valued mapping: it takes a vector $x \in V$ as input, and returns the set of vectors $\partial f(x)$ as output. When f is convex, the set-valued mapping ∂f is a "monotone," meaning that

$$\langle g - h, x - y \rangle \geq 0 \quad \text{whenever } x, y \in V \text{ and } g \in \partial f(x), h \in \partial f(y). \tag{2.2}$$

This fact turns out to be of fundamental importance.

If f is convex and differentiable at a point x in the interior of the domain of f, then it can be shown that $\partial f(x) = \{\nabla f(x)\}$. When f is differentiable on the entire space V, the

condition (2.2) reduces to the statement that $\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq 0$ for all $x, y \in V$. When $f : \mathbb{R} \to \mathbb{R}$ is convex and differentiable, condition (2.2) reduces to the statement that $(f'(x) - f'(y))(x - y) \geq 0$ for all $x, y \in \mathbb{R}$. This is equivalent to the statement that f' is monotone increasing: if $x \leq y$, then $f'(x) \leq f'(y)$.

Property (2.2) suggests a way to generalize much of the theory of convex functions: rather than studying convex functions, we can study set-valued mappings $F : V \to 2^V$ that satisfy the property

$$\langle g - h, x - y \rangle \geq 0 \quad \text{whenever } x, y \in V \text{ and } g \in F(x), h \in F(y). \tag{2.3}$$

A set-valued mapping F that satisfies this property is called "monotone." Rather than restricting our attention to the problem of finding x such that $0 \in \partial f(x)$, we can develop algorithms for solving more general "monotone inclusion problems" $0 \in F(x)$, where $F : V \to 2^V$ is monotone. Monotone mappings turn out to be quite well-behaved and have a rich theory. We make use of this theory throughout this thesis.

2.1.2 The convex conjugate

The basic idea behind duality in convex analysis is to view a closed convex set Ω as the intersection of all half spaces containing Ω. Applying this idea to epigraphs leads us to view a closed convex function as a supremum of affine functions.

Let V be a finite-dimensional inner product space over $\mathbb{R}$. A function $f : V \to \mathbb{R} \cup \{\infty\}$ might have many affine minorants with a given slope $y \in V$, but for the purpose of representing f we only need to consider the largest or "best" affine minorant with slope y. Note that

$$\begin{aligned} f(x) \geq \langle y, x \rangle - \beta \quad \forall\, x \in V &\iff \beta \geq \langle y, x \rangle - f(x) \quad \forall\, x \in V \\ &\iff \beta \geq \sup_{x \in V} \langle y, x \rangle - f(x). \end{aligned}$$

We see that the "best" choice of the constant β is

$$\beta = f^*(y) = \sup_{x \in V} \langle y, x \rangle - f(x). \tag{2.4}$$

The largest or "best" affine minorant of f with slope y is the function $g(x) = \langle y, x\rangle - f^*(y)$. The function f^* defined in equation (2.4) is called the "convex conjugate" of f.

The inequality $f(x) \geq \langle y, x\rangle - f^*(y)$, when it is written as

$$f(x) + f^*(y) \geq \langle y, x\rangle \quad \forall\, x, y \in V, \tag{2.5}$$

is known as Fenchel's inequality.

The fact that a closed convex set $\Omega \subset V$ is the intersection of all half spaces containing Ω, leads us to expect or to guess that a proper closed convex function $f : V \to \mathbb{R} \cup \{\infty\}$ is equal to the supremum of all affine minorants of f. In other words, we expect that

$$\begin{aligned} f(x) &= \sup_{y \in V} \langle y, x\rangle - f^*(y) \\ &= f^{**}(x) \end{aligned}$$

for all $x \in V$. This turns out to be true (see [Roc70] for example). This formula tells us how to recover f from its "dual representation" f^*. The fact that the inversion formula is so simple – you simply take the conjugate again – is surprising and mathematically beautiful.

Because f^* is a supremum of affine functions, we see that f^* is closed and convex (for any function $f : V \to \mathbb{R} \cup \{\infty\}$, not necessarily convex). If f is proper, then f^* is proper. Combining the above observations, we conclude that if $f : V \to \mathbb{R} \cup \{\infty\}$ is proper, then $f = f^{**}$ if and only if f is closed and convex.

Suppose $y \in \partial f(x)$. Then the "best" affine minorant of f with slope y is the one which is exact at x. Thus,

$$f(x) = \langle y, x\rangle - f^*(y). \tag{2.6}$$

On the other hand, if equation (2.6) is satisfied, then the affine minorant $\langle y, \cdot\rangle - f^*(y)$ is exact at x, and so $y \in \partial f(x)$. It follows that

$$y \in \partial f(x) \iff f(x) = \langle y, x\rangle - f^*(y).$$

In other words, $y \in \partial f(x)$ if and only if we have equality in the Fenchel inequality.

The Fenchel inequality (2.5), when written as $f^*(y) \geq \langle y, x\rangle - f(x)$, tells us that for any x the function $\langle \cdot, x\rangle - f(x)$ is an affine minorant of f^*. If $y \in \partial f(x)$, then this affine minorant is exact at y. Hence,

$$y \in \partial f(x) \implies x \in \partial f^*(y).$$

This result does not assume f is convex. If we have the additional assumption that f is proper closed convex, then

$$x \in \partial f^*(y) \implies y \in \partial f^{**}(x) = \partial f(x).$$

Therefore, if f is proper closed convex, then

$$y \in \partial f(x) \iff x \in \partial f^*(y).$$

The following calculus rules are also useful when working with convex conjugates.

Conjugate of indicator function Let f be the indicator function of a set $\Omega \subset V$. Then $f^*(z) = \sup_{x\in\Omega} \langle z, x\rangle$ is the support function of f.

Conjugate of a norm Let $f(x) = \|x\|$, where $\|\cdot\|$ is any norm on V. Then f^* is the indicator function of the dual norm unit ball.

Multiplication by scalar Let $f(x) = \lambda g(x)$. Then

$$\begin{aligned} f^*(z) &= \sup_{x\in V} \langle z, x\rangle - \lambda g(x) \\ &= \lambda \left(\sup_{x\in V} \langle z/\lambda, x\rangle - g(x) \right) \\ &= \lambda g^*(z/\lambda). \end{aligned}$$

The function $z \mapsto \lambda g^*(z/\lambda)$ is sometimes denoted $g^*\lambda$, and is called the "right scalar multiplication" of g^* by λ [Roc70, page 35]. With this notation, we have the elegant formula

$$(\lambda g)^* = g^*\lambda. \tag{2.7}$$

2.2 The dual problem and optimality conditions

To derive a dual problem for the canonical problem (1.1), we can first reformulate problem (1.1) as

$$\begin{aligned} \underset{x,y}{\text{minimize}} \quad & f(x) + g(y) \\ \text{subject to} \quad & Ax - y = 0. \end{aligned} \tag{2.8}$$

The Lagrangian for this reformulated problem is

$$L(x, y, z) = f(x) + g(y) + \langle z, Ax - y \rangle. \tag{2.9}$$

The dual function is

$$\begin{aligned} G(z) &= \inf_{x,y} L(x, y, z) \\ &= \inf_{x,y} f(x) - \langle -A^T z, x \rangle + g(y) - \langle z, y \rangle \\ &= -\inf_{x,y} \langle -A^T z, x \rangle - f(x) + \langle z, y \rangle - g(y) \\ &= -f^*(-A^T z) - g^*(z). \end{aligned}$$

The dual problem is to maximize $G(z)$. Expressed as a minimization problem, the dual problem is

$$\underset{z}{\text{minimize}} \quad f^*(-A^T z) + g^*(z). \tag{2.10}$$

The KKT optimality conditions for the canonical problem (1.1), which hold under a mild assumption on f, g and A, can be expressed concisely as a monotone inclusion problem. If x, y are optimal for the reformulated problem (2.8) and z is optimal for the dual problem (2.10), then the KKT conditions tell us that x and y are minimizers of the function $L(\cdot, \cdot, z)$. In other words, $0 \in \partial f(x) + A^T z$ and $0 \in \partial g(y) - z$. The second condition is equivalent to $y \in \partial g^*(z)$, which (using $y = Ax$) is equivalent to $0 \in -Ax + \partial g^*(z)$. In summary, we have

$$0 \in A^T z + \partial f(x), \qquad 0 \in -Ax + \partial g^*(z).$$

Using block notation, these conditions can be combined into the single inclusion

$$0 \in \begin{bmatrix} 0 & A^T \\ -A & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} \partial f(x) \\ \partial g^*(z) \end{bmatrix}. \tag{2.11}$$

The second term on the right denotes the Cartesian product $\partial f(x) \times \partial g^*(z)$. The vector on the left is added to the set on the right in the standard way. Note that this inclusion is a monotone inclusion, involving a sum of a skew-symmetric linear operator and a subdifferential operator. We will use these optimality conditions throughout this thesis.

Here is another way to derive the optimality condition (2.11). A vector x is optimal for problem (1.1) if and only if

$$0 \in \partial f(x) + A^T \partial g(Ax).$$

(Technically, a mild assumption is needed here to guarantee that $\partial(f + g \circ A)(x) = \partial f(x) + A^T \partial g(Ax)$.) In other words, x is optimal if and only if there exists $z \in \partial g(Ax)$ such that $0 \in \partial f(x) + A^T z$. But this is equivalent to the existence of z such that $Ax \in \partial g^*(z)$ and $0 \in A^T z + \partial f(x)$. In summary, x is optimal for problem (1.1) if and only if there exists z such that the inclusion (2.11) is satisfied.

We can also derive (2.11) by noting that finding primal and dual optimal variables (x, y) and z for (2.8) is equivalent to finding a saddle point of the Lagrangian (2.9). The variable y can be eliminated:

$$\inf_{x,y} L(x, y, z) = \inf_{x} \left(f(x) - g^*(z) + \langle z, Ax \rangle \right).$$

Thus our saddle point problem is reduced to finding a saddle point (x, z) of the convex-concave function

$$\ell(x, z) = f(x) - g^*(z) + \langle z, Ax \rangle.$$

The optimality conditions for this saddle point problem are

$$0 \in \partial f(x) + A^T z, \qquad 0 \in -Ax + \partial g^*(z). \tag{2.12}$$

The first condition in (2.12) states that x is a minimizer of $\ell(\cdot, z)$; the second condition in (2.12) states that z is a maximizer of $\ell(x, \cdot)$. As before, the optimality conditions (2.12) can be combined into the single condition (2.11) using block notation.

2.3 Monotone operators and resolvents

A multivalued or set-valued operator $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}^n$ maps points $x \in \mathbb{R}^n$ to sets $\mathcal{F}(x) \subset \mathbb{R}^n$. The domain of the operator is $\mathbf{dom}\,\mathcal{F} = \{x \mid \mathcal{F}(x) \neq \emptyset\}$ and its graph is the set $\{(x, y) \mid x \in \mathbf{dom}\,\mathcal{F}, y \in \mathcal{F}(x)\}$. The operator is *monotone* if

$$(y - \hat{y})^T(x - \hat{x}) \geq 0 \qquad \forall x,\ \hat{x},\ y \in \mathcal{F}(x),\ \hat{y} \in \mathcal{F}(\hat{x}).$$

A monotone operator is *maximal monotone* if its graph is not a strict subset of the graph of another monotone operator. The theory of monotone operators is discussed in depth in [BC11a].

2.3.1 Resolvent

The operator $(I + \lambda\mathcal{F})^{-1}$, where $\lambda > 0$, is called the *resolvent* of the operator $\mathcal{F}$. A fundamental result states that if F is maximal monotone, then $(I + \lambda\mathcal{F})^{-1}(x)$ exists and is unique for all $x \in \mathbb{R}^n$ [Bre73, proposition 2.2]. The value $u = (I + \lambda\mathcal{F})^{-1}(x)$ of the resolvent is the unique solution of the monotone inclusion

$$x \in u + \lambda\mathcal{F}(u). \tag{2.13}$$

The operators we will encounter in this thesis are combinations of two elementary types of maximal monotone operators. The first is the subdifferential ∂f of a closed convex function f with nonempty domain. The second type is a skew-symmetric linear operator of the form

$$\mathcal{F}(x, z) = \begin{bmatrix} 0 & C^T \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix}. \tag{2.14}$$

Other examples of monotone operators include saddle point operators

$$\mathcal{F}(x, y) = \begin{bmatrix} \partial_x f(x, y) \\ \partial_y(-f)(x, y) \end{bmatrix},$$

where f is a convex-concave function of x, y, and positive semidefinite linear operators

$$\mathcal{F}(x) = Ax$$

where A satisfies $\langle x, Ax\rangle \geq 0$ for all x (but A is not required to be symmetric).

The resolvent of a monotone (*i.e.*, positive semidefinite) linear operator $\mathcal{F}(x) = Ax$ is the matrix inverse $(I + \lambda A)^{-1}$. We note the following useful expressions for the resolvent of the skew-symmetric linear operator (2.14):

$$\begin{bmatrix} I & \lambda C^T \\ -\lambda C & I \end{bmatrix}^{-1} = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} + \begin{bmatrix} I \\ \lambda C \end{bmatrix} (I + \lambda^2 C^T C)^{-1} \begin{bmatrix} I \\ -\lambda C \end{bmatrix}^T \tag{2.15}$$

$$= \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} -\lambda C^T \\ I \end{bmatrix} (I + \lambda^2 C C^T)^{-1} \begin{bmatrix} \lambda C^T \\ I \end{bmatrix}^T. \tag{2.16}$$

This can be given a regularized least-squares interpretation. Let (x, z) be the value of the resolvent of $\mathcal{F}$,

$$\begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} I & \lambda C^T \\ -\lambda C & I \end{bmatrix}^{-1} \begin{bmatrix} \hat{x} \\ \hat{z} \end{bmatrix}.$$

Then, from (2.15), we see that $z = \hat{z} + \lambda C x$ and x is the solution of the least-squares problem

$$\text{minimize} \quad \|\lambda C x + \hat{z}\|^2 + \|x - \hat{x}\|^2.$$

Alternatively, from the second expression, $x = \hat{x} - \lambda C^T z$ and z is the solution of

$$\text{minimize} \quad \|\lambda C^T z - \hat{x}\|^2 + \|z - \hat{z}\|^2.$$

We now discuss the resolvents of subdifferential operators in more detail.

2.3.2 Proximal operator

If $\mathcal{F} = \partial f$, with f a closed convex function, the resolvent is also called the *proximal operator* or *prox-operator* of f and written $\text{prox}_{\lambda f} = (I + \lambda \partial f)^{-1}$. The prox-operator $\text{prox}_{\lambda f}$ maps x to the unique solution of the optimization problem

$$\text{minimize} \quad f(u) + \frac{1}{2\lambda}\|u - x\|^2 \tag{2.17}$$

with variable u. (Here $\|\cdot\|$ denotes Euclidean norm.) This can be seen by noting that the optimality condition for problem (2.17) is

$$\begin{aligned} 0 \in \frac{1}{\lambda}(u-x) + \partial f(u) &\iff x \in u + \lambda \partial f(u) \\ &\iff u = (I + \partial f)^{-1}(x). \end{aligned}$$

For example, the prox-operator of the indicator function of a nonempty closed convex set is the Euclidean projection on the set.

The following facts about proximal operators will be useful to us [CW05, CP07, BPC11].

Separable functions If f is separable, so that

$$f(x) = \sum_{i=1}^{k} f_i(x_i),$$

where x_i are subvectors of $x = (x_1, \ldots, x_k)$, then

$$\mathrm{prox}_f(x) = \begin{bmatrix} \mathrm{prox}_{f_1}(x_1) \\ \vdots \\ \mathrm{prox}_{f_k}(x_k) \end{bmatrix}. \tag{2.18}$$

Scaling of argument Suppose $f(x) = g(ax)$, where $a \in \mathbb{R}$, $a \neq 0$. Then

$$\mathrm{prox}_f(x) = \frac{1}{a}\mathrm{prox}_{a^2 g}(ax). \tag{2.19}$$

Moreau decomposition If $\lambda > 0$, then

$$x = \mathrm{prox}_{\lambda f}(x) + \lambda \mathrm{prox}_{\lambda^{-1} f^*}(x/\lambda) = \mathrm{prox}_{\lambda f}(x) + \mathrm{prox}_{f^* \lambda}(x). \tag{2.20}$$

Here, $f^*\lambda$ denotes the *right scalar multiplication* of f^* with λ, defined as $(f^*\lambda)(x) = \lambda f^*(x/\lambda)$ [Roc70, page 35]. (Right scalar multiplication was previously introduced in section 2.1.2.) It is easily verified that $(f^*\lambda) = (\lambda f)^*$ and $\mathrm{prox}_{f^*\lambda}(x) = \lambda \mathrm{prox}_{\lambda^{-1} f^*}(x/\lambda)$. This rule, known as the Moreau decomposition [Mor65], shows that the proximal operator of f^* (the convex conjugate of f) can be computed as easily as the proximal operator of f.

Here is a derivation of the Moreau decomposition: First let $u = \text{prox}_{f^*}(x)$. Then

$$\begin{aligned} x - u \in \partial f^*(u) &\implies u \in \partial f(x-u) \\ &\implies x \in x - u + \partial f(x-u) \\ &\implies x - u = \text{prox}_f(x) \\ &\implies x = \text{prox}_f(x) + \text{prox}_{f^*}(x). \end{aligned}$$

Equation (2.20) now follows by replacing f with λf and using the scaling rules (2.7) and (2.19).

In the case where f is the indicator function for a subspace W, it can be shown that f^* is the indicator function for $W^\perp$, the orthogonal complement of W. In this case, the Moreau decomposition reduces to the familiar fact that $x = P_W(x) + P_{W^\perp}(x)$.

Composition with affine mapping In general, computing the prox-operator of $f(x) = g(Ax+b)$ is not easy, even if the prox-operator of g can be evaluated efficiently. A notable exception, however, is the case where A is a matrix that satisfies $AA^T = (1/\alpha)I$ for some $\alpha > 0$. In this case we have

$$\text{prox}_f(x) = (I - \alpha A^T A)x + \alpha A^T(\text{prox}_{\alpha^{-1}g}(Ax+b) - b). \tag{2.21}$$

For future reference we also mention a few common examples, that can be proved directly from the definition or by using the properties listed above.

Quadratic function Suppose $f(x) = (1/2)\|Ax-b\|_2^2$, where $A \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m$, and $\lambda > 0$. Then

$$\begin{aligned} \text{prox}_{\lambda f}(x) &= (I + \lambda A^T A)^{-1}(x + \lambda A^T b) \\ &= (I - \lambda A^T(I + \lambda AA^T)^{-1}A)(x + \lambda A^T b). \end{aligned} \tag{2.22}$$

The second equality follows from the first by the matrix inversion lemma.

Indicator function Suppose $f(x)$ is the indicator function of a closed convex set C and $\lambda > 0$. Then $\mathrm{prox}_{\lambda f}(x)$ is the Euclidean projection $P_C(x)$ of x on C.

Norm Suppose $f(x)$ is a norm and $t > 0$. Then

$$\mathrm{prox}_{\lambda f}(x) = x - P_{\lambda C}(x) \tag{2.23}$$

where C is the unit ball of the dual norm. For $f(x) = \|x\|_1$, we have $\lambda C = \{x \mid -\lambda\mathbf{1} \leq x \leq \lambda\mathbf{1}\}$ and (2.23) reduces to *soft-thresholding:*

$$\mathrm{prox}_{\lambda f}(x)_k = \begin{cases} x_k - \lambda & x_k > \lambda \\ 0 & -\lambda \leq x_k \leq \lambda \\ x_k + \lambda & x_k \leq -\lambda. \end{cases}$$

If $f(x) = \|x\|$ is the Euclidean norm, then $P_{\lambda C}$ is projection on a ball with radius λ. We will also encounter the following pair of dual norms: for $(x, y) \in \mathbb{R}^n \times \mathbb{R}^n$,

$$\|(x,y)\|_{\mathrm{iso}} = \sum_{k=1}^{n} (x_k^2 + y_k^2)^{1/2}, \qquad \|(x,y)\|_{\mathrm{iso}*} = \max_{k=1,\ldots,n} (x_k^2 + y_k^2)^{1/2}. \tag{2.24}$$

(The subscript refers to the use of this norm to express isotropic 2D total variation; see section 3.3.) Here, the dual norm ball is a product of n two-dimensional Euclidean norm balls. Using (2.23) the prox-operator $(u, v) = \mathrm{prox}_{\lambda f}(x, y)$ of $f(x, y) = \|(x, y)\|_{\mathrm{iso}}$ can be computed as $(u_k, v_k) = \alpha_k(x_k, y_k)$ for $k = 1, \ldots, n$ with

$$\alpha_k = 1 - \frac{\lambda}{(x_k^2 + y_k^2)^{1/2}} \quad \text{if } (x_k^2 + y_k^2)^{1/2} > \lambda, \qquad \alpha_k = 0 \quad \text{otherwise.}$$

2.4 Douglas-Rachford splitting algorithm

In this section we provide a brief review of monotone operators and the Douglas-Rachford splitting algorithm. More details can be found in [EB92, BC11a, BPC11, PB13].

2.4.1 Douglas-Rachford splitting

The problem of finding a zero of a monotone operator $\mathcal{F}$, *i.e.*, solving $0 \in \mathcal{F}(x)$, is called a monotone inclusion problem. It is easily seen, for example from (2.13), that the zeros of a

maximal monotone operator $\mathcal{F}$ are the fixed points of the resolvent of $\mathcal{F}$. The fixed point iteration

$$x^k = (I + t\mathcal{F})^{-1}x^{k-1}, \quad k = 1, 2, \ldots,$$

is called the *proximal point algorithm* for solving the inclusion problem $0 \in \mathcal{F}(x)$ [Roc76b]. The proximal point algorithm converges under weak conditions (namely, $\mathcal{F}^{-1}(0) \neq \emptyset$ and maximal monotonicity of $\mathcal{F}$; see for example [EB92, theorem 3]), but is useful in practice only when the resolvent evaluations are inexpensive. Often this is not the case and one has to resort to algorithms based on operator splitting. A splitting algorithm decomposes the operator $\mathcal{F}$ as a sum $\mathcal{F} = \mathcal{A} + \mathcal{B}$ of two maximal monotone operators and requires only the resolvents of $\mathcal{A}$ or $\mathcal{B}$, or both.

A simple algorithm for solving $0 \in \mathcal{A}(x) + \mathcal{B}(x)$ is the Douglas-Rachford algorithm [LM79, Gab83, EB92, CP07]. The algorithm can be written in a number of equivalent forms, including

$$\begin{aligned} x^k &= (I + t\mathcal{A})^{-1}(z^{k-1}) && (2.25)\\ y^k &= (I + t\mathcal{B})^{-1}(2x^k - z^{k-1}) && (2.26)\\ z^k &= z^{k-1} + \rho(y^k - x^k). && (2.27) \end{aligned}$$

There are two algorithm parameters: a positive step size t and a relaxation parameter $\rho \in (0, 2)$. It can be shown that if $\mathcal{A}+\mathcal{B}$ has a zero, then z^k converges to a limit of the form $x+tv$, with $0 \in \mathcal{A}(x)+\mathcal{B}(x)$ and $v \in \mathcal{A}(x)\cap(-\mathcal{B}(x))$. Therefore $x^k = (I+t\mathcal{A})^{-1}(z^{k-1})$ converges to a solution x [EB92, BC11a]. Note that if $\mathcal{B}$ is linear, then the vector $r^k = (1/t)(I+t\mathcal{B})(x^k-y^k)$ satisfies

$$\begin{aligned} r^k &= \frac{1}{t}((I + t\mathcal{B})(x^k) - 2x^k + z^{k-1})\\ &= \mathcal{B}(x^k) + \frac{1}{t}(z^{k-1} - x^k)\\ &\in \mathcal{B}(x^k) + \mathcal{A}(x^k) \end{aligned}$$

and $r^k \to 0$ because $x^k - y^k \to 0$. We can therefore use a stopping criterion of the form $\|r^k\| \leq \epsilon$. The Douglas-Rachford method is useful if the resolvents of $\mathcal{A}$ and $\mathcal{B}$ are inexpensive,

compared to the resolvent of the sum $\mathcal{A} + \mathcal{B}$.

There is an equivalent form of the Douglas-Rachford algorithm that will be useful later when we derive the ADMM iteration. Let's take $\rho = 1$ for simplicity. We interchange $\mathcal{A}$ and $\mathcal{B}$ and start with the y-update (equation (2.26)):

$$y^k = (I + t\mathcal{A})^{-1}(2x^{k-1} - z^{k-1}), \quad z^k = z^{k-1} + y^k - x^{k-1}, \quad x^k = (I + t\mathcal{B})^{-1}(z^k).$$

We then switch the z and x updates:

$$y^k = (I + t\mathcal{A})^{-1}(2x^{k-1} - z^{k-1}), \quad x^k = (I + t\mathcal{B})^{-1}(z^{k-1} + y^k - x^{k-1}), \quad z^k = z^{k-1} + y^k - x^{k-1}.$$

Finally, we make a change of variables $w = z - x$:

$$y^k = (I + t\mathcal{A})^{-1}(x^{k-1} - w^{k-1}) \tag{2.28}$$

$$x^k = (I + t\mathcal{B})^{-1}(y^k + w^{k-1}) \tag{2.29}$$

$$w^k = w^{k-1} + y^k - x^k. \tag{2.30}$$

We now give a derivation of the Douglas-Rachford method. Our goal is to solve the monotone inclusion problem

$$0 \in \mathcal{A}(x) + \mathcal{B}(x), \tag{2.31}$$

where $\mathcal{A}$ and $\mathcal{B}$ are maximal monotone operators. We'll use the notation $R_{\mathcal{A}}$ and $R_{\mathcal{B}}$ to denote the resolvents of $\mathcal{A}$ and $\mathcal{B}$, respectively. In an attempt to make use of the resolvents of $\mathcal{A}$ and $\mathcal{B}$, let's write the inclusion (2.31) as

$$2x \in x + t\mathcal{A}(x) + x + t\mathcal{B}(x).$$

The vector x satisfies this inclusion if and only if there exists $z \in (I + t\mathcal{B})(x)$ such that

$$2x - z \in (I + t\mathcal{A})(x). \tag{2.32}$$

Noting that $x = (I + t\mathcal{B})^{-1}(z)$, we see that (2.32) implies that

$$2R_{\mathcal{B}}(z) - z \in (I + t\mathcal{A})(x). \tag{2.33}$$

On the left we recognize the Cayley operator of $\mathcal{B}$, denoted by $C_{\mathcal{B}}$. Inclusion (2.33) implies that

$$R_{\mathcal{A}}(C_{\mathcal{B}}(z)) = x = R_{\mathcal{B}}(z),$$

which implies that

$$2R_{\mathcal{A}}(C_{\mathcal{B}}(z)) - C_{\mathcal{B}}(z) = 2R_{\mathcal{B}}(z) - C_{\mathcal{B}}(z) = z,$$

which is equivalent to the fixed point equation

$$C_{\mathcal{A}}(C_{\mathcal{B}}(z)) = z. \tag{2.34}$$

($C_{\mathcal{A}}$ is the Cayley operator of $\mathcal{A}$.) The Douglas-Rachford method solves this fixed point equation using damped fixed point iteration. In other words, we solve the equivalent equation

$$\left(\frac{1}{2}I + \frac{1}{2}C_{\mathcal{A}} \circ C_{\mathcal{B}}\right)(z) = z \tag{2.35}$$

using fixed point iteration. The Cayley operators $C_{\mathcal{A}}$ and $C_{\mathcal{B}}$ are nonexpansive, which implies that the operator $(1/2)I + (1/2)C_{\mathcal{A}} \circ C_{\mathcal{B}}$ is firmly nonexpansive, which guarantees that the fixed point iteration

$$z^k = \left(\frac{1}{2}I + \frac{1}{2}C_{\mathcal{A}} \circ C_{\mathcal{B}}\right)(z^{k-1})$$

will converge to a solution of (2.35) (assuming a solution exists).

(To say that an operator T is firmly nonexpansive means that $\langle T(x) - T(y), x - y\rangle \geq \|T(x) - T(y)\|^2$ for all x, y. For more details on monotone operator theory, including a convergence proof for fixed point iteration with a firmly nonexpansive operator, see [BC11a, EB92, DY14].)

Let's examine the steps needed to carry out this fixed point iteration. Given z^{k-1}, we first compute $x^k = R_{\mathcal{B}}(z^{k-1})$. We then compute $y^k = R_{\mathcal{A}}(2x^k - z^{k-1})$. Finally we compute

$$\begin{aligned} z^k &= \frac{z^{k-1}}{2} + \frac{2y^k - (2x^k - z^{k-1})}{2} \\ &= z^{k-1} + y^k - x^k. \end{aligned}$$

These are the steps of the Douglas-Rachford iteration.

2.4.2 Spingarn's method

The Douglas-Rachford splitting method can be used to minimize a sum $f(x) + g(x)$ of two convex functions by taking $\mathcal{A} = \partial f$ and $\mathcal{B} = \partial g$. In this case, the resolvents in the algorithm (2.25)–(2.27) are prox-operators, $(I + t\mathcal{A})^{-1} = \mathrm{prox}_{tf}$ and $(I + t\mathcal{B})^{-1} = \mathrm{prox}_{tg}$, and the method can be viewed as alternating between the two minimization problems that define these prox-operators. Two well-known algorithms, Spingarn's method of partial inverses [Spi83, Spi85] and the alternating direction method of multipliers (ADMM) [GM75, GM76], can be interpreted as applications of this idea.

Spingarn's method of partial inverses [Spi83, Spi85] is a method for minimizing a convex function f over a subspace $\mathcal{V}$,

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \mathcal{V}. \end{array}$$

Eckstein and Bertsekas [EB92] have shown that the method is equivalent to the Douglas-Rachford method for minimizing $f(x) + \delta_{\mathcal{V}}(x)$, where $\delta_{\mathcal{V}}$ is the indicator function of $\mathcal{V}$. Each iteration in the algorithm requires an evaluation of the prox-operator of f and a Euclidean projection on the subspace $\mathcal{V}$.

Spingarn's method can be used to solve the canonical problem (1.1) by reformulating the problem as

$$\begin{array}{ll} \underset{x,y}{\text{minimize}} & f(x) + g(y) \\ \text{subject to} & (x, y) \in \mathcal{V}, \end{array}$$

where $\mathcal{V} = \{(x, y) \mid Ax - y = 0\}$. Evaluating the prox-operator of the objective function $f(x) + g(y)$ reduces to evaluating the prox-operators of f and g separately, and projecting onto $\mathcal{V}$ is equivalent to projecting onto the null space of the matrix $\begin{bmatrix} A & -I \end{bmatrix}$, which requires solving a linear system with coefficient $I + A^T A$.

2.4.3 Alternating minimization

The alternating direction method of multipliers [GM75, GM76] is a dual decomposition method for problems of the form

$$\begin{array}{ll} \text{minimize} & f(x) + g(y) \\ \text{subject to} & Ax + By = c. \end{array}$$

The method is also known as the split Bregman method [GO09]. As shown in [Gab83, EB92], the algorithm can be interpreted as the Douglas-Rachford method applied to the dual problem

$$\text{minimize} \quad c^T z + f^*(-A^T z) + g^*(-B^T z), \tag{2.36}$$

with the dual objective split as a sum of two convex functions $c^T z + f^*(-A^T z)$ and $g^*(-B^T z)$. After a series of simplifications, the Douglas-Rachford iteration (with $\rho = 1$) can be written as

$$\begin{aligned} x^k &= \operatorname*{argmin}_x L(x, y^{k-1}, z^{k-1}) \\ y^k &= \operatorname*{argmin}_y L(x^k, y, z^{k-1}) \qquad (2.37) \\ z^k &= z^{k-1} + t(Ax^k + By^k - c) \qquad (2.38) \end{aligned}$$

where L is the augmented Lagrangian

$$L(x, y, z) = f(x) + g(y) + z^T(Ax + By - c) + \frac{t}{2}\|Ax + By - c\|^2.$$

To improve convergence one can modify the basic ADMM algorithm to include overrelaxation [EB92, BPC11]. In the overrelaxed version of ADMM, we replace the expression Ax^k in (2.37) and (2.38) with $\rho Ax^k - (1 - \rho)(By^{k-1} - c)$ where $\rho \in (0, 2)$ [BPC11, page 21]. For more details on ADMM we refer the reader to the recent surveys [BPC11, Eck12].

Here we will derive the ADMM iteration by applying Douglas-Rachford to the dual problem (2.36). Solving (2.36) is equivalent to solving the monotone inclusion problem $0 \in \mathcal{A}(z) + \mathcal{B}(z)$, where $\mathcal{A}$ is the subdifferential of $F(z) = f^*(-A^T z) + \langle z, c \rangle$ and $\mathcal{B}$ is the

subdifferential of $G(z) = g^*(-B^T z)$. We will use the version (2.28)-(2.30) of the Douglas-Rachford iteration, rewritten here with different variable names (because the vector we want to solve for is now called z, not x):

$$\begin{aligned} p^k &= (I + t\mathcal{A})^{-1}(z^{k-1} - w^{k-1}) \\ z^k &= (I + t\mathcal{B})^{-1}(p^k + w^{k-1}) \\ w^k &= w^{k-1} + p^k - z^k. \end{aligned}$$

Under mild assumptions, the iterates z^k are guaranteed to converge to a solution of the dual problem (2.36) (assuming a solution exists).

In the first step we evaluate the resolvent of $\mathcal{A}$, which is the proximal operator of F. We will now show that

$$\text{prox}_{tF}(\hat{z}) = \hat{z} + t(Ax - c) \tag{2.39}$$

where

$$x = \underset{u}{\text{argmin}} \quad f(u) + \frac{t}{2}\left\|Au - c + \frac{\hat{z}}{t}\right\|^2. \tag{2.40}$$

(Any choice of $x \in \text{argmin}_u\ f(u) + (t/2)\|Au - c + \hat{z}/t\|^2$ gives the same value of $\text{prox}_{tF}(\hat{z})$.) To see this, observe that

$$\begin{aligned} z = \text{prox}_{tF}(\hat{z}) &\implies \frac{\hat{z} - z}{t} \in \partial F(z) \\ &\implies z \in \hat{z} + t(A\partial f^*(-A^T z) - c) \\ &\implies z = \hat{z} + t(Ax - c) \end{aligned}$$

for some x such that

$$\begin{aligned} x \in \partial f^*(-A^T z) &\implies -A^T z \in \partial f(x) \\ &\implies 0 \in \partial f(x) + A^T z \\ &\implies 0 \in \partial f(x) + A^T(\hat{z} + t(Ax - c)) \\ &\implies x \in \underset{u}{\text{argmin}}\, f(u) + \frac{t}{2}\left\|Ax - c + \frac{\hat{z}}{t}\right\|^2. \end{aligned}$$

We can reverse these steps to show that if $x \in \operatorname{argmin}_u f(u) + (t/2)\|Au - c + \hat{z}/t\|^2$, then $\operatorname{prox}_{tF}(\hat{z}) = \hat{z} + t(Ax - c)$. A similar calculation allows us to evaluate the resolvent of $\mathcal{B}$.

Note that F is the dual function for the problem

$$\begin{array}{ll} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & Ax = c, \end{array}$$

and in equation (2.40) we are minimizing the augmented Lagrangian for this problem. Formulas (2.39) – (2.40) for the prox-operator of F show that the augmented Lagrangian method is equivalent to solving the dual problem via the proximal point method.

Using (2.39) to evaluate prox-operators, the Douglas-Rachford iteration becomes

$$x^k = \operatorname*{argmin}_u f(u) + \frac{t}{2}\left\|Au - c + \frac{z^{k-1} - w^{k-1}}{t}\right\|^2 \tag{2.41}$$

$$p^k = z^{k-1} - w^{k-1} + t(Ax^k - c) \tag{2.42}$$

$$\begin{aligned} y^k &= \operatorname*{argmin}_u g(u) + \frac{t}{2}\left\|Bu + \frac{z^{k-1}}{t} - \frac{w^{k-1}}{t} + Ax^k - c + \frac{w^{k-1}}{t}\right\|^2 \\ &= \operatorname*{argmin}_u g(u) + \frac{t}{2}\left\|Ax^k + Bu - c + \frac{z^{k-1}}{t}\right\|^2 \\ z^k &= z^{k-1} - w^{k-1} + t(Ax^k - c) + w^{k-1} + tBy^k \\ &= z^{k-1} + t(Ax^k + By^k - c) \\ w^k &= w^{k-1} + p^k - z^k \\ &= z^{k-1} + t(Ax^k - c) - z^{k-1} - t(Ax^k + By^k - c) \\ &= -tBy^k. \end{aligned} \tag{2.43}$$

Notice that the equation (2.42) can be omitted. From equation (2.43), we have $-w^k/t = By^k$. Replacing $-w^{k-1}/t$ with By^{k-1} in equation (2.41), the Douglas-Rachford iteration reduces

to

$$
\begin{aligned}
x^k &= \operatorname*{argmin}_u \; f(u) + \frac{t}{2}\left\| Au + By^{k-1} - c + \frac{z^{k-1}}{t} \right\|^2 \\
y^k &= \operatorname*{argmin}_u \; g(u) + \frac{t}{2}\left\| Ax^{k} + Bu - c + \frac{z^{k-1}}{t} \right\|^2 \\
z^k &= z^{k-1} + t(Ax^k + By^k - c).
\end{aligned}
$$

This is the ADMM iteration.

It's also possible to derive the Douglas-Rachford iteration for minimizing $f(x) + g(x)$ (where f, g are proper closed convex) by applying ADMM to the equivalent problem

$$
\begin{aligned}
\underset{x,y}{\text{minimize}} \quad & f(x) + g(y) \\
\text{subject to} \quad & x - y = 0.
\end{aligned}
\tag{2.44}
$$

The augmented Lagrangian for problem (2.44) is

$$
L(x, y, z) = f(x) + g(y) + \langle z, x - y \rangle + \frac{t}{2}\left\| x - y \right\|^2.
$$

Upon completing the square in the augmented Lagrangian, we see that the ADMM iteration for this problem is

$$
\begin{aligned}
x^k &= \operatorname*{argmin}_x \; f(x) + \frac{t}{2}\left\| x - y^{k-1} + \frac{z^{k-1}}{t} \right\|^2 \\
&= \operatorname{prox}_{(1/t)f}\left(y^{k-1} - \frac{z^{k-1}}{t} \right), \\
y^k &= \operatorname*{argmin}_y \; g(y) + \frac{t}{2}\left\| x^{k} - y + \frac{z^{k-1}}{t} \right\|^2 \\
&= \operatorname{prox}_{(1/t)g}\left(x^{k} + \frac{z^{k-1}}{t} \right), \\
z^k &= z^{k-1} + t(x^k - y^k).
\end{aligned}
$$

The last equation can be written as $z^k/t = z^{k-1}/t + x^k - y^k$. Making a change of variable $w^k = z^k/t$, this ADMM iteration is seen to be equivalent to the Douglas-Rachford iteration (2.28) - (2.30) with step size $1/t$.

CHAPTER 3

Primal-dual operator splitting[1]

3.1 Introduction

We discuss primal-dual splitting methods for convex optimization problems of the form

$$\text{minimize} \quad f(x) + g(Ax), \tag{3.1}$$

where f and g are closed convex functions with inexpensive proximal operators and A is a structured matrix. Specifically, we distinguish two types of matrix structure. In the simplest case, structure in A makes it possible to solve equations with a coefficient matrix $I + \lambda A^T A$, where $\lambda > 0$, efficiently. In the second and more general case, A can be decomposed as a sum $A = B + C$ of two structured matrices, with the same meaning of 'structure': linear equations with coefficients $I + \lambda B^T B$ and $I + \lambda C^T C$ can be solved efficiently. These assumptions are motivated by applications in image processing. A variety of image deblurring problems can be formulated as large convex optimization problems of the form (3.1), with f and g simple convex penalty functions or indicator functions of simple convex sets. The matrix A represents the blurring operation and the linear transformations used in regularization terms (see section 3.3). The first of the two types of structure mentioned above arises when periodic boundary conditions are used. In this case $A^T A$ can be diagonalized by multiplication with discrete Fourier transform matrices, so equations with coefficient matrix $I + \lambda A^T A$ are solved efficiently via fast Fourier transforms. The second type of structure arises when more realistic boundary conditions (for example, replicate boundary conditions) are used. In the sum $A = B + C$ the first term then represents the model assuming periodic boundary

[1] **This chapter is based on the paper [OV14].**

conditions; the second term is a sparse term added to correct for the non-periodic boundary conditions. Linear equations with coefficients $I + \lambda B^T B$ and $I + \lambda C^T C$ can therefore be solved efficiently, via the fast Fourier transform and sparse matrix factorization methods, respectively. However, these techniques are not easily combined in an efficient method for inverting $I + \lambda A^T A$.

The algorithms we propose use the Douglas-Rachford splitting method applied to systems of primal-dual optimality conditions. This primal-dual approach is interesting for several reasons. In contrast to purely primal or dual applications of the Douglas-Rachford method (such as the alternating direction method of multipliers or ADMM) it does not require a reformulation of the problem and the introduction of a potentially large number of auxiliary variables or constraints. An advantage over forward-backward splitting algorithms or semi-implicit primal-dual methods (such as the Chambolle-Pock algorithm) is that the selection of suitable step sizes is easier and is not limited by the norms of the linear operators. We test the performance of the primal-dual splitting method on a set of image deblurring examples. In each experiment the primal-dual method is compared with three other methods: ADMM, the Douglas-Rachford method applied to a reformulated primal problem, and the Chambolle-Pock method. The results indicate that the convergence of the primal-dual approach is comparable or better than the other methods, although all perform well with properly chosen algorithm parameters.

This chapter is organized as follows. In section 3.2 we derive several primal-dual splitting strategies for the optimality conditions of (3.1), under the different assumptions on the structure in A. In the second half of this chapter the results are applied to image deblurring problems with various types of convex regularization and constraints. In section 3.3 we first describe the image deblurring problem and formulate it as a convex optimization problem of the form (3.1). We then present five examples. In section 3.4 the primal-dual splitting method is applied to constrained total variation deblurring problems with a non-quadratic fidelity term. In section 3.5 we discuss image restoration problems with a tight frame regularization. In section 3.6 we consider an application to image deblurring with a spatially

varying, piecewise-constant blurring operator. This chapter concludes with a summary of the main points in section 3.7.

3.2 Primal-dual splitting strategies

We now apply the Douglas-Rachford splitting method to develop decomposition algorithms for the optimization problem (3.1). This problem format is widely used as a standard form in the literature on multiplier and splitting methods (see, for example, [GM76, Gab83]). Its popularity derives from the fact that large-scale problems in practical applications can often be expressed in this form with relatively simple choices for f, g, and A.

Note that there is flexibility in the choice of g and A. Replacing the problem with

$$\text{minimize} \quad f(x) + \tilde{g}(\tilde{A}x), \tag{3.2}$$

where $\tilde{A} = \beta A$ and $\tilde{g}(y) = g(y/\beta)$ does not change the optimal solution x or the optimal value of the problem. This observation will allow us to introduce an additional algorithm parameter that can be adjusted to improve convergence.

3.2.1 Optimality conditions

The optimality conditions for (3.1) can be written as

$$0 \in \begin{bmatrix} 0 & 0 & A^T & I \\ 0 & 0 & -I & 0 \\ -A & I & 0 & 0 \\ -I & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} + \begin{bmatrix} 0 \\ \partial g(y) \\ 0 \\ \partial f^*(w) \end{bmatrix}. \tag{3.3}$$

(The second term on the right denotes the set $\{0\} \times \partial g(y) \times \{0\} \times \partial f^*(w)$.) The variable y is an auxilary variable in the reformulated primal problem

$$\begin{aligned} &\text{minimize} && f(x) + g(y) \\ &\text{subject to} && Ax - y = 0. \end{aligned} \tag{3.4}$$

The variable z is the dual multiplier for the constraint $Ax = y$. The variable w is a variable in the dual problem, written as

$$\begin{array}{ll} \text{maximize} & -f^*(w) - g^*(z) \\ \text{subject to} & A^T z + w = 0. \end{array}$$

Several equivalent reduced forms of the optimality conditions can be obtained by eliminating y (using the relation $(\partial g)^{-1} = \partial g^*$ between the subdifferential of a function and its conjugate), by eliminating w (using $(\partial f^*)^{-1} = \partial f$), or eliminating y and w. The first two of these options lead to 3×3 block systems

$$0 \in \begin{bmatrix} 0 & A^T & I \\ -A & 0 & 0 \\ -I & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ z \\ w \end{bmatrix} + \begin{bmatrix} 0 \\ \partial g^*(z) \\ \partial f^*(w) \end{bmatrix} \tag{3.5}$$

$$0 \in \begin{bmatrix} 0 & 0 & A^T \\ 0 & 0 & -I \\ -A & I & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} \partial f(x) \\ \partial g(y) \\ 0 \end{bmatrix}. \tag{3.6}$$

Eliminating y and w results in a 2×2 system

$$0 \in \begin{bmatrix} 0 & A^T \\ -A & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} \partial f(x) \\ \partial g^*(z) \end{bmatrix}. \tag{3.7}$$

3.2.2 Simple splitting

The most straightforward primal-dual splitting approach is to write the right-hand side of the 2×2 system (3.7) as a sum of the two monotone operators

$$\mathcal{A}(x, z) = \begin{bmatrix} \partial f(x) \\ \partial g^*(z) \end{bmatrix}, \qquad \mathcal{B}(x, z) = \begin{bmatrix} 0 & A^T \\ -A & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix}.$$

The resolvents of the two operators were given in section 2.3. The value $(x, z) = (I + t\mathcal{A})^{-1}(\hat{x}, \hat{z})$ of the resolvent of $\mathcal{A}$ is given by

$$x = \text{prox}_{tf}(\hat{x}), \qquad z = \text{prox}_{tg^*}(\hat{z}).$$

Using the Moreau decomposition the second proximal operator can be written in alternate form as

$$z = \hat{z} - t\mathrm{prox}_{t^{-1}g}(\hat{z}/t) = \hat{z} - \mathrm{prox}_{gt}(\hat{z})$$

(where gt denotes right scalar multiplication of g). The resolvent of $\mathcal{B}$ is a linear mapping

$$\begin{aligned}(I + t\mathcal{B})^{-1} &= \begin{bmatrix} I & tA^T \\ -tA & I \end{bmatrix}^{-1} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} + \begin{bmatrix} I \\ tA \end{bmatrix} (I + t^2 A^T A)^{-1} \begin{bmatrix} I \\ -tA \end{bmatrix}^T.\end{aligned}$$

An iteration of the Douglas-Rachford algorithm applied to $\mathcal{A} + \mathcal{B}$ is therefore as follows:

$$x^k = \mathrm{prox}_{tf}(p^{k-1}) \tag{3.8}$$

$$z^k = \mathrm{prox}_{tg^*}(q^{k-1}) \tag{3.9}$$

$$\begin{bmatrix} u^k \\ v^k \end{bmatrix} = \begin{bmatrix} I & tA^T \\ -tA & I \end{bmatrix}^{-1} \begin{bmatrix} 2x^k - p^{k-1} \\ 2z^k - q^{k-1} \end{bmatrix} \tag{3.10}$$

$$p^k = p^{k-1} + \rho(u^k - x^k) \tag{3.11}$$

$$q^k = q^{k-1} + \rho(v^k - z^k). \tag{3.12}$$

The simple splitting strategy is useful when the prox-operators of f and g are inexpensive, and the structure in A allows us to solve linear equations with coefficient $I + t^2 A^T A$ efficiently.

It is interesting to work out the differences when the same method is applied to the scaled problem (3.2), *i.e.*, when g is replaced with $\tilde{g}(y) = g(y/\beta)$ and A with $\tilde{A} = \beta A$. After a few simplifications in notation, the Douglas-Rachford iteration for the scaled problem can

be written in terms of the original g and A as follows:

$$\begin{aligned} x^k &= \mathrm{prox}_{tf}(p^{k-1}) \\ \tilde{z}^k &= \mathrm{prox}_{sg^*}(\tilde{q}^{k-1}) \\ \begin{bmatrix} u^k \\ \tilde{v}^k \end{bmatrix} &= \begin{bmatrix} I & tA^T \\ -sA & I \end{bmatrix}^{-1} \begin{bmatrix} 2x^k - p^{k-1} \\ 2\tilde{z}^k - \tilde{q}^{k-1} \end{bmatrix} \\ p^k &= p^{k-1} + \rho(u^k - x^k) \\ \tilde{q}^k &= \tilde{q}^{k-1} + \rho(\tilde{v}^k - \tilde{z}^k). \end{aligned}$$

The parameter s is given by $s = \beta^2 t$, and can be interpreted as a dual step size, which can be chosen arbitrarily and independently of the (primal) step size t.

Primal Douglas-Rachford splitting (Spingarn's method) The type of structure exploited in the primal-dual 'simple splitting' method is also easily handled by the Douglas-Rachford method applied to the primal or the dual problem. In the primal application, we approach the reformulated problem (3.4) as one of minimizing a separable function $f(x)+g(y)$ over a subspace $\mathcal{V} = \{(x, y) \mid y = Ax\}$. The primal Douglas-Rachford method (or Spingarn's method) applied to this problem therefore involves evaluations of the prox-operator of $f(x) + g(y)$, *i.e.*, the prox-operators of f and g, and Euclidean projection on $\mathcal{V}$. The projection (x, y) of a vector $(\hat{x}, \hat{y})$ on $\mathcal{V}$ can be expressed as

$$x = (I + A^T A)^{-1}(\hat{x} + A^T \hat{y}), \qquad y = Ax.$$

The steps in the primal Douglas-Rachford method are therefore very similar (but not equivalent) to (3.8)–(3.12). In particular, the complexity per iteration is the same.

Dual Douglas-Rachford splitting (ADMM) In the dual application of the Douglas-Rachford method, better known as ADMM, one first reformulates the problem by introducing

a 'dummy' variable u and adding a constraint $u = x$:

$$\begin{array}{ll} \text{minimize} & f(u) + g(y) \\ \text{subject to} & \begin{bmatrix} I \\ A \end{bmatrix} x - \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} u \\ y \end{bmatrix} = 0. \end{array} \tag{3.13}$$

The augmented Lagrangian of this problem is

$$f(u) + g(y) + w^T(x-u) + z^T(Ax-y) + \frac{t}{2}\left(\|x-u\|^2 + \|Ax-y\|^2\right).$$

In ADMM, one alternates between minimization of the augmented Lagrangian over x and over (u, y). Minimization over x is a least-squares problem with Hessian matrix $I + A^TA$. Minimization over u, y requires evaluations of the prox-operators of f and g. One iteration can be written as follows:

$$\begin{aligned} x^k &= (I + A^TA)^{-1}\left(u^{k-1} + A^Ty^{k-1} - \frac{1}{t}(w^{k-1} + A^Tz^{k-1})\right) \\ u^k &= \text{prox}_{t^{-1}f}(x^k + w^{k-1}/t) \\ y^k &= \text{prox}_{t^{-1}g}(Ax^k + z^{k-1}/t) \\ w^k &= w^{k-1} + t(x^k - u^k) \\ z^k &= z^{k-1} + t(Ax^k - y^k). \end{aligned}$$

As can be seen, the per-iteration complexity is the same as in the primal-dual and primal methods. One difference is that ADMM requires the introduction of an additional variable u. If the residual in the equality constraint $x = u$ is slow to decrease to zero, this can be expected to impact the convergence of x. We will return to this issue in section 3.2.5.

Chambolle-Pock method Another interesting method for (3.1) is the Chambolle-Pock algorithm [CP11a]. One iteration of the Chambolle-Pock method consists of the following steps:

$$\begin{aligned} x^k &= \text{prox}_{tf}(x^{k-1} - tA^Tz^{k-1}) \\ z^k &= \text{prox}_{tg^*}(z^{k-1} + tA(2x^k - x^{k-1})). \end{aligned}$$

This algorithm has the important advantage that it does not require the solution of linear equations, but only multiplications with A and A^T. However, convergence of the algorithm depends on the step size t, which must be chosen in $(0, 1/\|A\|)$, where $\|A\|$ is the maximum singular value of A. As shown in [CP11a] the method can be interpreted as a preconditioned ADMM. When one of the functions f or g^* is strongly convex, an accelerated version of Chambolle-Pock can be used. Since the strong convexity assumption does not hold for the applications studied in this chapter, we omit the details.

Different primal and dual step sizes t and s can be used in the Chambolle-Pock method. With different step sizes the iteration is

$$\begin{aligned} x^k &= \text{prox}_{tf}(x^{k-1} - tA^T z^{k-1}) \\ z^k &= \text{prox}_{sg^*}(z^{k-1} + sA(2x^k - x^{k-1})). \end{aligned}$$

Convergence is guaranteed if $\sqrt{st} < 1/\|A\|$. This variation of the Chambolle-Pock method can also be interpreted as the basic version of the method (with a single step size t), applied to the scaled problem (3.2). The dual step size s and the scale factor β are related by $s = \beta^2 t$. As a further improvement, one can apply overrelaxation (see [Con13] for details). The iteration for the overrelaxed version is

$$\begin{aligned} \bar{x}^k &= \text{prox}_{tf}(x^{k-1} - tA^T z^{k-1}) \\ \bar{z}^k &= \text{prox}_{sg^*}(z^{k-1} + sA(2\bar{x}^k - x^{k-1})) \\ (x^k, z^k) &= \rho(\bar{x}^k, \bar{z}^k) + (1-\rho)(x^{k-1}, z^{k-1}) \end{aligned}$$

where $\rho \in (0, 2)$.

3.2.3 Mixed splitting

We now consider problem (3.1) under weaker assumptions on the problem structure. As before, we assume that f and g have inexpensive prox-operators. In addition we assume A can be decomposed as $A = B + C$ where B and C have the property that equations with coefficients $I + \lambda B^T B$ and $I + \lambda C^T C$, with $\lambda > 0$, are easy to solve (but not necessarily

equations with coefficient $I+\lambda A^T A$). We apply the Douglas-Rachford method to the primal-dual optimality condition (3.3), with the right-hand side decomposed as the sum of two monotone operators

$$\begin{aligned}
\mathcal{A}(x,y,z,w) &= \begin{bmatrix} 0 & 0 & B^T & 0 \\ 0 & 0 & 0 & 0 \\ -B & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} + \begin{bmatrix} 0 \\ \partial g(y) \\ 0 \\ \partial f^*(w) \end{bmatrix} \\
\mathcal{B}(x,y,z,w) &= \begin{bmatrix} 0 & 0 & C^T & I \\ 0 & 0 & -I & 0 \\ -C & I & 0 & 0 \\ -I & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}.
\end{aligned}$$

The resolvents of $\mathcal{A}$ and $\mathcal{B}$ can be evaluated as follows. The value $(x,y,z,w) = (I + t\mathcal{A})^{-1}(\hat{x},\hat{y},\hat{z},\hat{w})$ of the resolvent of $\mathcal{A}$ is the solution of the inclusion problem

$$\begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \\ \hat{w} \end{bmatrix} \in \begin{bmatrix} I & 0 & tB^T & 0 \\ 0 & I & 0 & 0 \\ -tB & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} + t \begin{bmatrix} 0 \\ \partial g(y) \\ 0 \\ \partial f^*(w) \end{bmatrix}.$$

The computation of y, w, and (x,z) can be separated. The solution for y and w is

$$y = \mathrm{prox}_{tg}(\hat{y}), \qquad w = \mathrm{prox}_{tf^*}(w).$$

The solution x, z follows from

$$\begin{bmatrix} x \\ z \end{bmatrix} = \begin{bmatrix} I & tB^T \\ -tB & I \end{bmatrix}^{-1} \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = \begin{bmatrix} 0 \\ \hat{y} \end{bmatrix} + \begin{bmatrix} I \\ tB \end{bmatrix} (I + t^2 B^T B)^{-1}(\hat{x} - tB\hat{y})$$

(using the expression (2.15)). The resolvent of $\mathcal{B}$ is a linear operator. It can be verified that

$$
\begin{aligned}
(I+t\mathcal{B})^{-1} \;=\; & \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{1}{1+t^2}I & \frac{t}{1+t^2}I & 0 \\ 0 & -\frac{t}{1+t^2}I & \frac{1}{1+t^2}I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \\
& + \begin{bmatrix} I \\ \frac{t^2}{1+t^2}C \\ \frac{t}{1+t^2}C \\ tI \end{bmatrix} \left((1+t^2)I + \frac{t^2}{1+t^2}C^TC \right)^{-1} \begin{bmatrix} I \\ \frac{t^2}{1+t^2}C \\ -\frac{t}{1+t^2}C \\ -tI \end{bmatrix}^T .
\end{aligned}
$$

From these expressions, we see that the resolvents of $\mathcal{A}$ and $\mathcal{B}$ require proximal operators of f and g, and linear equations with coefficients of the form $I+\lambda B^TB$ and $I+\lambda C^TC$. These operations are inexpensive, under our assumptions on the problem structure, and therefore each iteration of the Douglas-Rachford method is relatively cheap.

Primal Douglas-Rachford splitting (Spingarn's method) As one of the referees pointed out, the additive structure in $A = B + C$ can be handled in the primal Douglas-Rachford method, at the cost of doubling the number of variables, via the reformulation

$$
\begin{array}{ll} \text{minimize} & f(x_1) + g(y_1+y_2) + \delta(x_1 - x_2) \\ \text{subject to} & Bx_1 - y_1 = 0 \\ & Cx_2 - y_2 = 0 \end{array} \tag{3.14}
$$

where δ is the indicator function of $\{0\}$. The variables in the reformulated problem are x_1, x_2, y_1, y_2. The cost function and its proximal-operator are separable in two sets of variables (x_1, x_2) and (y_1, y_2). The proximal operator of $f(x_1) + \delta(x_1 - x_2)$, as a function of x_1 and x_2, reduces to an evaluation of the prox-operator of f. The prox-operator of $g(y_1 + y_2)$ as a function of y_1 and y_2 can be related to the prox-operator of g via the formula (2.21). The Euclidean projection (x_1, x_2, y_1, y_2) of $(\hat{x}_1, \hat{x}_2, \hat{y}_1, \hat{y}_2)$ on the subspace defined by the constraints is given by

$$
x_1 = (I + B^TB)^{-1}(\hat{x}_1 + B^T\hat{y}_1), \qquad x_2 = (I + C^TC)^{-1}(\hat{x}_2 + C^T\hat{y}_2),
$$

and $y_1 = Bx_1$, $y_2 = Cx_2$.

Dual Douglas-Rachford splitting (ADMM) A similar reformulation, with an extra dummy variable u, brings the problem in a form amenable to ADMM:

$$\begin{array}{ll} \text{minimize} & f(u) + g(y_1 + y_2) \\ \text{subject to} & \begin{bmatrix} I & 0 \\ 0 & I \\ B & 0 \\ 0 & C \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} I & 0 & 0 \\ I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} u \\ y_1 \\ y_2 \end{bmatrix} = 0. \end{array}$$

One iteration of ADMM involves an alternating minimization of the augmented Lagrangian over (x_1, x_2), and (u, y_1, y_2). This requires evaluations of the prox-operators of f and g, and the solution of least-squares problems with Hessian matrices $I + B^T B$ and $I + C^T C$.

Chambolle-Pock method The Chambolle-Pock method described in section 3.2.2 only involves multiplications with A and A^T and therefore applies without modification when A can be decomposed as $A = B + C$ with structured B and C.

3.2.4 Mixed splitting with partially quadratic functions

The mixed splitting strategy can be simplified if f or g are simple quadratic functions, or separable functions that include quadratic terms. We illustrate this with two examples.

Quadratic f Supppose f in (3.1) is quadratic, of the form $f(x) = (\mu/2)x^T x + a^T x$ with $\mu \geq 0$. In that case we can start from the optimality conditions in the simpler 3×3 form (3.6).

Since $\partial f(x) = \{\mu x + a\}$, we can use the splitting

$$\begin{aligned}
\mathcal{A}(x,y,z) &= \begin{bmatrix} \mu I & 0 & B^T \\ 0 & 0 & 0 \\ -B & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} a \\ \partial g(y) \\ 0 \end{bmatrix} \\
\mathcal{B}(x,y,z) &= \begin{bmatrix} 0 & 0 & C^T \\ 0 & 0 & -I \\ -C & I & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}.
\end{aligned}$$

The resolvent of $\mathcal{A}$ maps $(\hat{x}, \hat{y}, \hat{z})$ to the solution (x, y, z) of the inclusion

$$\begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{bmatrix} \in \begin{bmatrix} (1+\mu t)I & 0 & tB^T \\ 0 & I & 0 \\ -tB & 0 & I \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + t \begin{bmatrix} a \\ \partial g(y) \\ 0 \end{bmatrix}.$$

The solution is $y = \mathrm{prox}_{tg}(\hat{y})$ and

$$\begin{aligned}
\begin{bmatrix} x \\ z \end{bmatrix} &= \begin{bmatrix} (1+\mu t)I & tB^T \\ -tB & I \end{bmatrix}^{-1} \begin{bmatrix} \hat{x} - ta \\ \hat{z} \end{bmatrix} \\
&= \left(\begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} + \begin{bmatrix} I \\ tB \end{bmatrix} \left((1+\mu t)I + t^2 B^T B \right)^{-1} \begin{bmatrix} I \\ -tB \end{bmatrix}^T \right) \begin{bmatrix} \hat{x} - ta \\ \hat{z} \end{bmatrix}.
\end{aligned}$$

The operator $\mathcal{B}$ is linear so its resolvent is a matrix inverse

$$\begin{aligned}
(I + t\mathcal{B})^{-1} &= \frac{1}{1+t^2} \begin{bmatrix} 0 & 0 & 0 \\ 0 & I & tI \\ 0 & -tI & I \end{bmatrix} \\
&\quad + \begin{bmatrix} I \\ \frac{t^2}{1+t^2} C \\ \frac{t}{1+t^2} C \end{bmatrix} \left(I + \frac{t^2}{1+t^2} C^T C \right)^{-1} \begin{bmatrix} I \\ \frac{t^2}{1+t^2} C \\ -\frac{t}{1+t^2} C \end{bmatrix}^T.
\end{aligned}$$

The Douglas-Rachford method applied to $\mathcal{A} + \mathcal{B}$ is therefore of interest under the same assumptions as in the general mixed splitting case of section 3.2.3: the function g has an

inexpensive proximal operator, and the matrices B and C possess structure that allow us to solve linear equations with coefficients of the form $I + \lambda B^T B$ and $I + \lambda C^T C$ fast. However, since the number of variables in the 3×3 system (3.6) is smaller, one can expect the Douglas-Rachford method to converge faster than for the general mixed splitting described in section 3.2.3.

Quadratic g A similar simplification is possible when g is quadratic, of the form $g(y) = \|y - a\|^2/(2\mu)$, for $\mu > 0$. In this case we start from the optimality conditions (3.5) and note that $g^*(z) = (\mu/2)z^T z + a^T z$ and $\partial g^*(z) = \{\mu z + a\}$. We choose the splitting

$$\begin{aligned}
\mathcal{A}(x,z,w) &= \begin{bmatrix} 0 & B^T & 0 \\ -B & \mu I & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ z \\ w \end{bmatrix} + \begin{bmatrix} 0 \\ a \\ \partial f^*(w) \end{bmatrix} \\
\mathcal{B}(x,z,w) &= \begin{bmatrix} 0 & C^T & I \\ -C & 0 & 0 \\ -I & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ z \\ w \end{bmatrix}.
\end{aligned}$$

The resolvent of $\mathcal{A}$ maps $(\hat{x}, \hat{z}, \hat{w})$ to the solution of

$$\begin{bmatrix} \hat{x} \\ \hat{z} \\ \hat{w} \end{bmatrix} \in \begin{bmatrix} I & tB^T & 0 \\ -tB & (1+\mu t)I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} x \\ z \\ w \end{bmatrix} + t \begin{bmatrix} 0 \\ a \\ \partial f^*(w) \end{bmatrix}.$$

The solution is $w = \mathrm{prox}_{tf^*}(\hat{w})$ and

$$\begin{aligned}
\begin{bmatrix} x \\ z \end{bmatrix} &= \begin{bmatrix} I & tB^T \\ -tB & (1+\mu t)I \end{bmatrix}^{-1} \begin{bmatrix} \hat{x} \\ \hat{z} - ta \end{bmatrix} \\
&= \left(\frac{1}{1+\mu t} \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} + \begin{bmatrix} I \\ \frac{t}{1+\mu t} B \end{bmatrix} \left(I + \frac{t^2}{1+\mu t} B^T B \right)^{-1} \begin{bmatrix} I \\ -\frac{t}{1+\mu t} B \end{bmatrix}^T \right) \begin{bmatrix} \hat{x} \\ \hat{z} - ta \end{bmatrix}.
\end{aligned}$$

The resolvent of $\mathcal{B}$ is

$$(I + t\mathcal{B})^{-1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} + \begin{bmatrix} I \\ tC \\ tI \end{bmatrix} \left((1+t^2)I + t^2 C^T C\right)^{-1} \begin{bmatrix} I \\ -tC \\ -tI \end{bmatrix}^T.$$

Evaluating the two resolvents requires an evaluation of the proximal operator of f^*, a linear equation with coefficient of the form $I + \lambda B^T B$, and a linear equation $I + \lambda C^T C$.

Partially quadratic functions More generally, one can simplify the general mixed splitting method whenever f is separable, of the form $f(x_1, x_2) = f_1(x_1) + f_2(x_2)$ with f_1 a simple quadratic term, or g is separable, $g(y_1, y_2) = g_1(y_1) + g_2(y_2)$ with g_1 quadratic. The details are straightforward and the idea will be illustrated with an example in section 3.5.2.

3.2.5 Example

The main advantage of the primal-dual splitting method is that it exploits the problem structure without introducing extra variables and constraints. This can be expected to benefit the speed of convergence and accuracy attained. The primal and dual Douglas-Rachford methods on the other hand have the same complexity per iteration as the primal-dual method, but only after a reformulation of the problem which requires extra variables and constraints. We also note that the increase in problem dimensions is greater in the dual than in the primal approach. Unfortunately a theoretical analysis of this effect is lacking and it is not clear how important it is in practice. To conclude this section we therefore give a small experiment that illustrates the difference.

We compare the mixed splitting approaches of section 3.2.3 on the test problem

$$\text{minimize} \quad \|x\| + \gamma \|(B + C)x - b\|_1$$

with $n = 500$ variables and square matrices B, C. This problem has the form (3.1) with $f(x) = \|x\|$ and $g(y) = \gamma\|y - b\|_1$, so the algorithms of section 3.2.3 can be applied. The problem data are generated randomly, with the components of A, B, b drawn independently

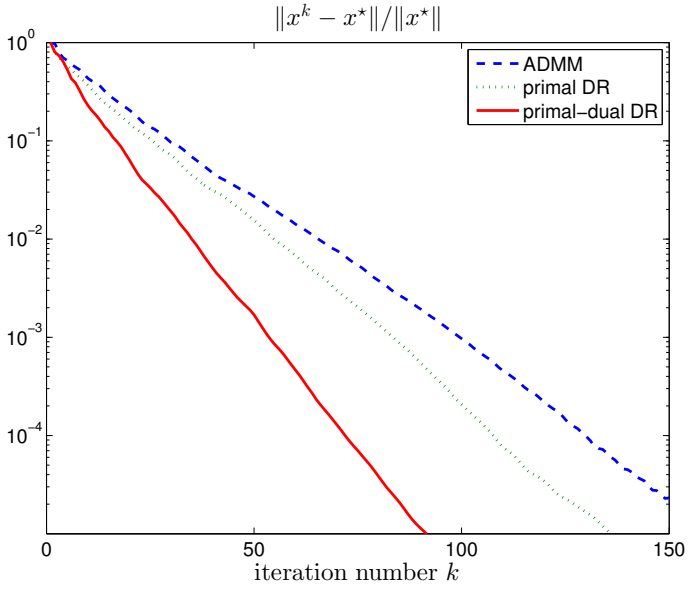


Figure 3.1: Relative error versus iteration number for the experiment in section 3.2.5.

from a standard normal distribution, $C = A - B$, and $\gamma = 1/100$. In figure 3.1 we compare the convergence of the primal-dual mixed splitting method, the primal Douglas-Rachford method, and ADMM (dual Douglas-Rachford method). The relative error $\|x^k - x^\star\|/\|x^\star\|$ is with respect to the solution $x^\star$ computed using CVX [GB12, GB08]. For each method, the three algorithm parameters (primal and dual step sizes, and overrelaxation parameter) were tuned by trial and error to give fastest convergence. As can be seen, the primal-dual splitting method shows a clear advantage on this problem class. We also note that ADMM is slightly slower than the primal Douglas-Rachford method, which is consistent with the intuition that fewer auxiliary variables and constraints is better.

3.3 Image deblurring by convex optimization

In the second half of this chapter we apply the primal-dual splitting methods to image deblurring. The blurring model, and how to express the deblurring problem in the canonical form (1.1), are discussed in section 1.2.

When the functions ϕ_{f}, ϕ_{r}, ϕ_{s} introduced in section 1.2 are quadratic (squared Euclidean

norms), the deblurring problem (1.4) reduces to a linear equation of the form

$$(K^T K + \rho I + \sigma D^T D)x = K^T b.$$

If the matrices K and D can be diagonalized by multiplication with DFT or DCT matrices, fast Fourier transform methods can be used to solve the equation in $O(N^2 \log N)$ operations [Vog02, HNO06].

When one or more of the terms in the objective for problem (1.4) is not quadratic, the problem must be solved by an iterative optimization algorithm, customized to exploit the structure in K and D. Second-order methods, such as Newton's method or interior-point methods [CGM99] [Vog02, chapter 8] are limited because they require the solution of linear equations $K^T H_{\mathrm{f}} K + H_{\mathrm{r}} + D^T H_{\mathrm{s}} D$ where the matrices H_{f}, H_{r}, H_{s} are multiples of the Hessians of ϕ_{f}, ϕ_{r}, ϕ_{s} (if these functions are twice differentiable) or positive definite matrices that result from block-elimination of the Newton equations in an interior-point method. The presence of these matrices makes it difficult to exploit structure in K and D. Iterative linear equation solvers can be applied, using the FFT for matrix-vector multiplications, but the reduced accuracy of iterative equation solvers often impairs the fast convergence of Newton's method or the interior-point method.

Most published deblurring algorithms were developed for special cases of problem (1.4), such as the TV-regularized deblurring problem (1.5). Several of the existing algorithms of TV-regularized deblurring can be generalized in a straightforward manner to the general problem. Two such classes of methods are the ADMM and Chambolle-Pock algorithms introduced in the previous section. The ADMM or split Bregman method described in section 3.2.2 is a popular dual decomposition algorithm for large-scale convex optimization [GM75, GM76, Gab83, GO09, BPC11, CTY13]. As illustrated in [BPC11] it can be used to derive simple, efficient algorithms for a range of applications. To apply ADMM to problem (1.4) (in its most general case, with non-quadratic ϕ_{f}, ϕ_{r}, ϕ_{s}), one first rewrites the problem in a form suited for dual decomposition, *i.e.*, with a separable objective and linear

equality constraints,

$$\begin{array}{ll} \text{minimize} & \phi_{\mathrm{f}}(v) + \phi_{\mathrm{r}}(u) + \phi_{\mathrm{s}}(w) \\ \text{subject to} & u = x \\ & v = Kx - b \\ & w = Dx, \end{array}$$

where u, v, w are auxiliary variables. The main step in each ADMM iteration is an alternating minimization of the augmented Lagrangian for this problem,

$$\begin{aligned} L(x,u,v,w) &= \phi_{\mathrm{f}}(v) + \phi_{\mathrm{r}}(u) + \phi_{\mathrm{s}}(w) \\ &\quad + \frac{t}{2}\left(\|u - x + \frac{1}{t}p\|^2 + \|v - Kx + b + \frac{1}{t}q\|^2 + \|w - Dx + \frac{1}{t}r\|^2 \right). \end{aligned}$$

In this expression t is a positive step size, and p, q, and r denote multipliers for the three constraints. The first of the two alternating minimization steps is over x, keeping u, v, w constant; the second is over (u, v, w) with fixed x. Since L is separable in u, v, w, the second step involves three independent minimizations. The minimizations over u, v, w are straightforward (often with complexity $O(N^2)$) if the functions ϕ_{f}, ϕ_{f}, and ϕ_{s} are simple, as will be the case in the applications considered here. The minimization over x involves a linear equation with coefficient matrix $I + K^T K + D^T D$. If K and D are diagonalizable by multiplication with DFT or DCT matrices, this can be reduced to a diagonal equation and solved in $O(N^2 \log N)$ operations. The total cost of one iteration of the ADMM method is therefore $O(N^2 \log N)$.

A second important and versatile class of first-order algorithms contains the Chambolle-Pock algorithm and the related primal-dual methods developed in [EZC09, Ess10, BC11b, HY12, Con13]. These methods solve the primal-dual optimality conditions

$$0 \in \begin{bmatrix} 0 & K^T & D^T \\ -K & 0 & 0 \\ -D & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} \partial\phi_{\mathrm{r}}(x) \\ b + \partial\phi_{\mathrm{f}}^*(y) \\ \partial\phi_{\mathrm{s}}^*(z) \end{bmatrix}. \tag{3.15}$$

by a semi-implicit forward-backward iteration which requires matrix-vector multiplications with K and D and their transposes, but not the solution of any linear equations. Several

other types of modified forward-backward splitting algorithms can be applied to the primal-dual optimality conditions, with a similar complexity per iteration as the Chambolle-Pock method [CP12, Tse00].

3.4 Total variation deblurring

In this section we present two applications of the primal-dual splitting methods to a constrained L_1 TV deblurring problem

$$\begin{array}{ll} \text{minimize} & \|Kx - b\|_1 + \gamma\|Dx\|_{\text{iso}} \\ \text{subject to} & 0 \leq x \leq \mathbf{1} \end{array} \tag{3.16}$$

where b is a blurry, noisy image with n pixels stored as a vector, K represents a convolution operator and D represents a discrete gradient operator. The variable is an n-vector x. This problem can be written in the canonical form (3.1) with

$$f(x) = \delta_S(x), \qquad g(y_1, y_2) = \|y_1 - b\|_1 + \gamma\|y_2\|_{\text{iso}}, \qquad A = \begin{bmatrix} K \\ D \end{bmatrix}, \tag{3.17}$$

where $S = \{x \mid 0 \leq x \leq \mathbf{1}\}$.

All the experiments were performed on a computer with a 3.00 GHz AMD Phenom(tm) II X4 945 processor with 4 cores and 8 GB of RAM. The code was written in MATLAB using MATLAB version 8.1.0.604 (R2013a).

3.4.1 Periodic boundary conditions

In the first experiment, periodic boundary conditions are used in the definitions of K and D. The matrices K^TK and D^TD can therefore be diagonalized by the discrete Fourier basis matrix, and equations with coefficient

$$I + t^2A^TA = I + t^2K^TK + t^2D^TD$$

are solved very efficiently via fast Fourier transforms. We can therefore apply the algorithms given in section 3.2.2.

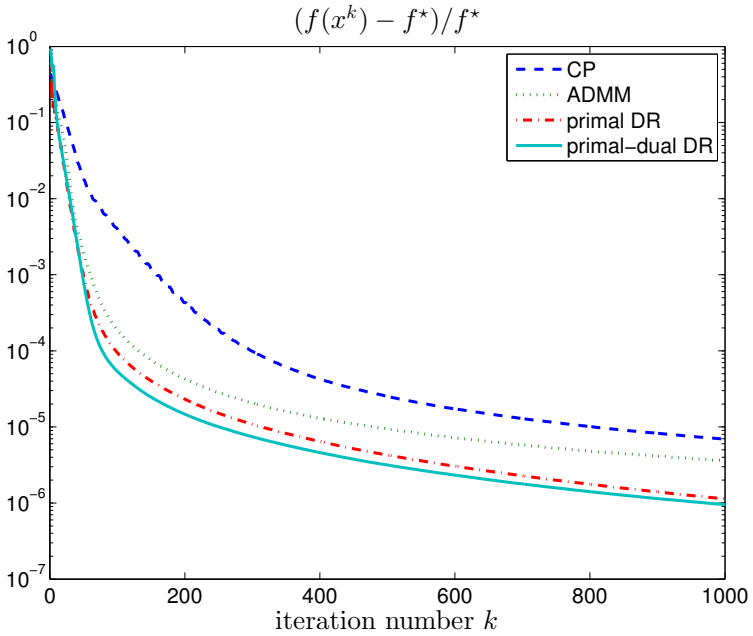


Figure 3.2: Relative optimality gap versus iteration number for the experiment in section 3.4.1.

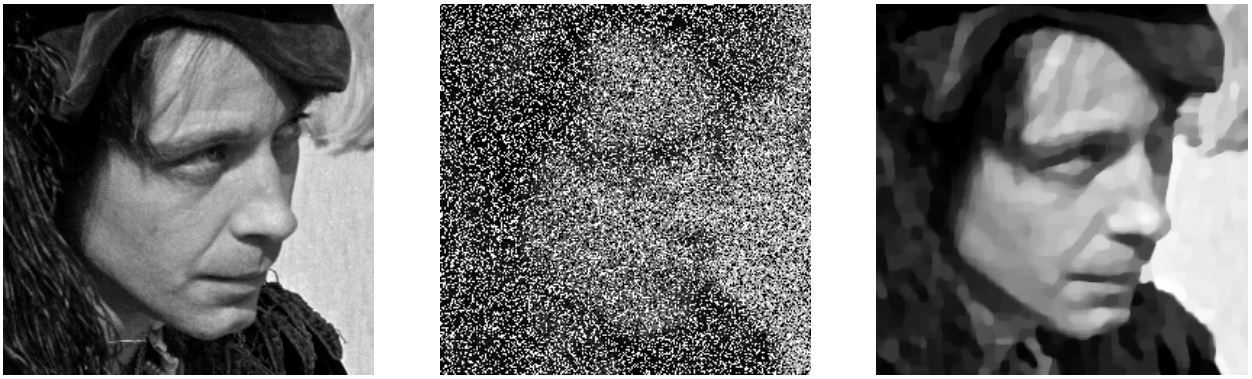
Figure 3.2 compares the performances of the Chambolle-Pock, ADMM, primal Douglas-Rachford, and primal-dual Douglas-Rachford algorithms. The test image b is a degraded version of the 1024 by 1024 "Man" image from the USC-SIPI Image Database[2]. The original image (scaled so that intensity values are between 0 and 1) was blurred with a 15 by 15 truncated Gaussian kernel with standard deviation $\sigma = 7$. Then salt and pepper noise was added to a random selection of 50% of the pixels. The parameter γ was chosen to give a visually appealing image reconstruction. A nearly optimal primal objective value $f^\star$ was computed by running the primal-dual Douglas-Rachford algorithm for $10,000$ iterations. In figure 3.2 the quantity $(f^k - f^\star)/f^\star$ is plotted against the iteration number k, where f^k is the primal objective value at iteration k. The original, blurry/noisy, and restored (by primal-dual DR) images are shown in figures 3.3 and 3.4.

For each method, close to optimal fixed primal and dual step sizes (and overrelaxation parameters) were selected by trial and error. The Chambolle-Pock step sizes s and t were

[2] `http://sipi.usc.edu/database/`

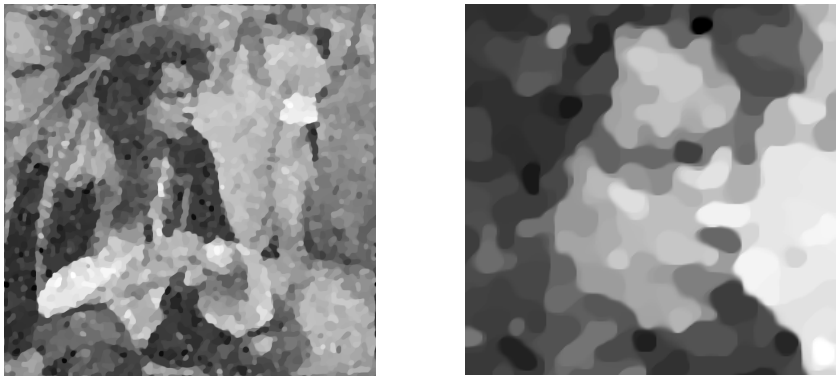
(a) Original image. (b) Blurry, noisy image. (c) Restored image.

Figure 3.3: Result for the experiment in section 3.4.1.



(a) Close-up on original image. (b) Close-up on degraded image. (c) Close-up on restored image.

Figure 3.4: Close-ups of the results for the experiment in section 3.4.1.



(a) Deblurred using L_2 data fidelity term.

(b) Close-up on restoration using L_2 data fidelity term.

Figure 3.5: Result of using L_2 data fidelity term for salt and pepper noise.

chosen to satisfy $tsL^2 = 1$, where $L = (\|K\|^2 + \|D\|^2)^{1/2}$ is an upper bound on $\|A\|$. Note that the norms of K and D can be computed analytically because $K^T K$ and $D^T D$ are diagonalized by the discrete Fourier basis matrix. The average elapsed time per iteration was 1.37 seconds for Chambolle-Pock, 1.33 seconds for ADMM, 1.33 seconds for primal DR, and 1.46 seconds for primal-dual DR.

As can be seen from the convergence plots, the four methods reach a modest accuracy quickly. After a few hundred iterations, progress slows down considerably. In this example the algorithms based on Douglas-Rachford converge faster than the Chambolle-Pock algorithm. The time per iteration is roughly the same for each method and is dominated by 2D fast Fourier transforms.

The quality of the restored image is good because the L_1 data fidelity is very well suited to deal with salt and pepper noise. Using an L_2 data fidelity term and omitting the interval constraints leads to a much poorer result. To illustrate this, figure 3.5 shows the result of minimizing $\|Kx - b\|^2 + \gamma\|Dx\|_{\text{iso}}$, with γ chosen to give the most visually appealing reconstruction.

3.4.2 Non-periodic boundary conditions

To illustrate the methods of section 3.2.3, we consider the same problem (3.16) with non-periodic boundary conditions [CYP05, Sor12]. Now K represents a convolution operator that uses replicate boundary conditions and D represents a discrete gradient operator that uses symmetric boundary conditions. The matrices K and D can be decomposed as $K = K_{\mathrm{p}} + K_{\mathrm{s}}$ and $D = D_{\mathrm{p}} + D_{\mathrm{s}}$ where K_{p} and D_{p} represent convolution operators that use periodic boundary conditions, and K_{s} and D_{s} are sparse matrices that correct the values near the boundary. Correspondingly, the matrix A in (3.17) can be decomposed as $A = B + C$, where

$$B = \begin{bmatrix} K_{\mathrm{p}} \\ D_{\mathrm{p}} \end{bmatrix}, \qquad C = \begin{bmatrix} K_{\mathrm{s}} \\ D_{\mathrm{s}} \end{bmatrix}.$$

The two resolvents needed in the mixed splitting algorithm of section 3.2.3 are inexpensive to evaluate: in the resolvent of $\mathcal{A}$ one exploits the fact that $B^T B = K_{\mathrm{p}}^T K_{\mathrm{p}} + D_{\mathrm{p}}^T D_{\mathrm{p}}$ can be diagonalized by the discrete Fourier transform basis; the resolvent of $\mathcal{B}$ involves a linear equation with a sparse matrix $C^T C = K_{\mathrm{s}}^T K_{\mathrm{s}} + D_{\mathrm{s}}^T D_{\mathrm{s}}$.

Figure 3.6 compares the performances of the Chambolle-Pock, ADMM, primal Douglas-Rachford, and primal-dual Douglas-Rachford algorithms when b is a degraded version of the 256 by 256 "cameraman" image. The original image (scaled so that intensity values are between 0 and 1) was blurred with a 9 by 9 truncated Gaussian kernel with standard deviation $\sigma = 4$. Then salt and pepper noise was added to a random selection of 10% of the pixels. The parameter γ was chosen to give a visually appealing image reconstruction. A nearly optimal primal objective value $f^\star$ was computed by running the primal-dual Douglas-Rachford algorithm for $10,000$ iterations. In figure 3.6 the quantity $(f^k - f^\star)/f^\star$ is plotted against the iteration number k. (f^k is the primal objective value at iteration k.) The blurry/noisy and restored (by primal-dual DR) images are shown in figure 3.7.

Also shown in figure 3.7 is a restoration (by primal-dual DR) using periodic boundary conditions (and all other parameters unchanged). In this example we see the value of using correct boundary conditions, as the quality of the restoration with incorrect boundary

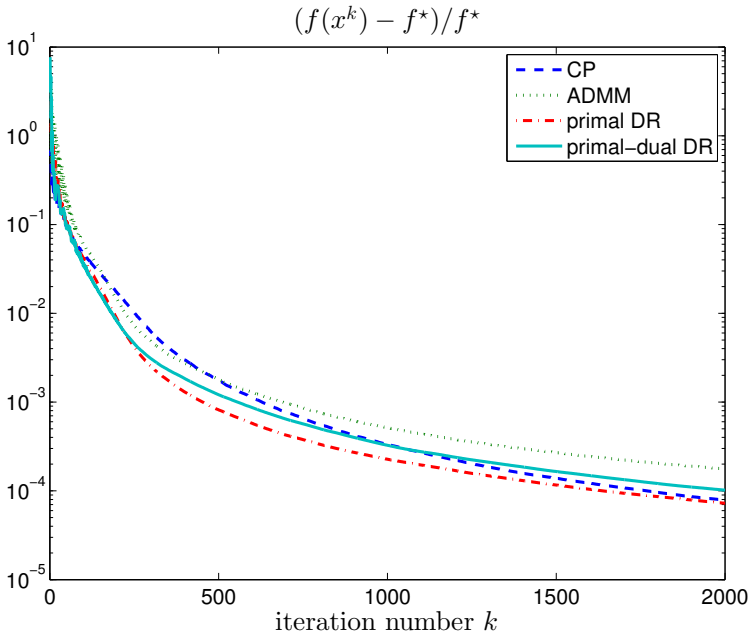


Figure 3.6: Relative optimality gap vs. iteration for the experiment in section 3.4.2.

(a) Blurred with 9 by 9 kernel, 10% salt and pepper noise.

(b) Restored using periodic boundary conditions.

(c) Restored using replicate boundary conditions.

Figure 3.7: Result for the experiment in section 3.4.2.

conditions is poor.

For all methods, close to optimal fixed primal and dual step sizes (and overrelaxation parameters) were selected by trial and error. Primal and dual step sizes were implemented in the algorithms based on Douglas-Rachford by modifying g and A as in equation (3.2). The Chambolle-Pock step sizes s and t were chosen to satisfy $tsL^2 = 1$, where $L = (\|K_{\mathrm{p}}\|^2 + \|D_{\mathrm{p}}\|^2)^{1/2} \approx \|A\|$. The norms of K_{p} and D_{p} can be computed analytically because $K_{\mathrm{p}}^T K_{\mathrm{p}}$ and $D_{\mathrm{p}}^T D_{\mathrm{p}}$ are diagonalized by the discrete Fourier basis. The average elapsed time per iteration was 0.08 seconds for Chambolle-Pock, 0.12 seconds for ADMM, 0.10 seconds for primal Douglas-Rachford, and 0.10 seconds for primal-dual Douglas-Rachford. In this example, the modification to Chambolle-Pock (incorporating an overrelaxation step) mentioned in section 3.2 allows Chambolle-Pock to be competitive with the algorithms based on Douglas-Rachford.

3.5 Tight frame regularized deblurring

In the next two experiments, we use tight frame regularization rather than TV regularization in the deblurring model (1.4). We take $\phi_{\mathrm{s}}(Dx) = \gamma\|Dx\|_1$, with D a tight frame analysis operator, *i.e.*, $D^T D = \alpha I$ for some positive α [Mal99, chapter 5] [KL12].

3.5.1 Periodic boundary conditions

We first consider

$$\begin{array}{ll} \text{minimize} & \|Kx - b\|_1 + \gamma\|Dx\|_1 \\ \text{subject to} & 0 \leq x \leq \mathbf{1} \end{array} \tag{3.18}$$

where, as before, b is a blurry, noisy image stored as an n-vector and K represents a convolution operator constructed using periodic boundary conditions. The matrix D is the analysis operator for a (shearlet) tight frame [KL12]. The MATLAB package Shearlab-1.1 is used to evaluate the tight operator [KSZ11].

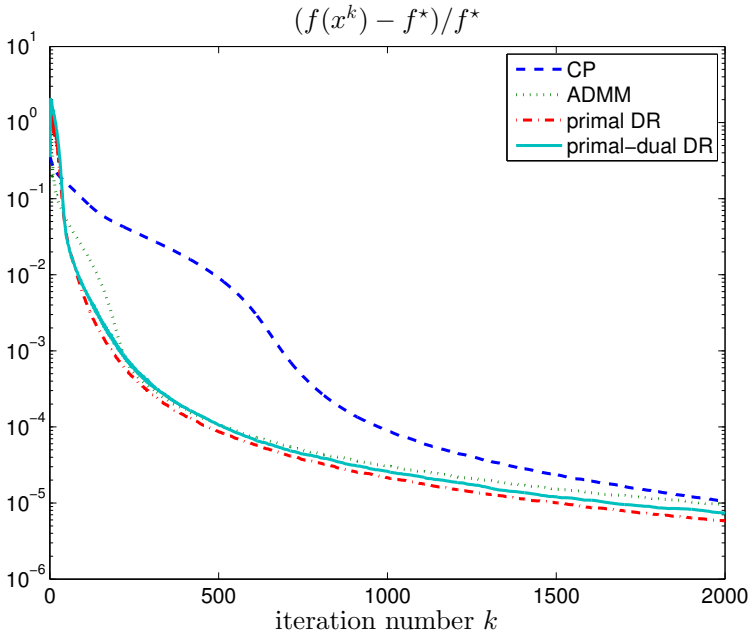


Figure 3.8: Relative optimality gap vs. iteration for the experiment in section 3.5.1.

The problem (3.18) is a special case of problem (3.1) with

$$f(x) = \delta_S(x), \qquad g(y_1, y_2) = \|y_1 - b\|_1 + \gamma\|y_2\|_1 \qquad A = \begin{bmatrix} K \\ D \end{bmatrix},$$

and $S = \{x \mid 0 \leq x \leq \mathbf{1}\}$. Because K^TK and $D^TD = \alpha I$ are both diagonalized by the discrete Fourier basis, we can use the simple splitting algorithm given in section 3.2.2.

Figure 3.8 compares the Chambolle-Pock, primal Douglas-Rachford, ADMM, and primal-dual Douglas-Rachford algorithms. The test image b is a degraded version of the 256 by 256 "cameraman" image. The original image (scaled so that intensity values are between 0 and 1) was blurred with a 7 by 7 truncated Gaussian kernel with standard deviation $\sigma = 5$. Then salt and pepper noise was added to a random selection of 30% of the pixels. The parameter γ was chosen to give a visually appealing image reconstruction. A nearly optimal primal objective value $f^\star$ was computed by running the primal-dual Douglas-Rachford algorithm for $10,000$ iterations. In figure 3.8 the quantity $(f^k - f^\star)/f^\star$ is plotted against the iteration number k, where f^k is the primal objective value at iteration k. The original, blurry/noisy, and restored (by primal-dual splitting) images are shown in figure 3.9.

(a) Original image (b) Blurry, noisy image (c) Restored image

Figure 3.9: Result for the experiment in section 3.5.1.

For each method, close to optimal fixed primal and dual step sizes (and overrelaxation parameters) were selected by trial and error. The Chambolle-Pock step sizes s and t were chosen to satisfy $tsL^2 = 1$, where $L = (\|K\|^2 + \|D\|^2)^{1/2}$ is an upper bound on $\|A\|$. Note that the norm of K can be computed analytically because $K^T K$ is diagonalized by the discrete Fourier basis matrix. The norm of D is $\sqrt{\alpha}$ because $D^T D = \alpha I$. The average elapsed time per iteration was 0.54 seconds for Chambolle-Pock, 0.51 seconds for ADMM, 0.51 seconds for primal DR, and 0.50 seconds for primal-dual DR.

The convergence plots are similar to the previous experiment (in section 3.4.1). As is typical for first-order methods, a modest accuracy is reached quickly but progress slows down after the first few hundred iterations. In this example the algorithms based on Douglas-Rachford converge faster than the Chambolle-Pock algorithm. The time per iteration is roughly the same for each method and is dominated by 2D fast Fourier Transforms and shearlet transforms.

3.5.2 Non-periodic boundary conditions

We solve a problem similar to (3.18), but with a quadratic fidelity term and without the constraints on x:

$$\text{minimize} \quad \frac{1}{2}\|Kx - b\|^2 + \gamma\|Dx\|_1. \tag{3.19}$$

We will use replicate boundary conditions for K. As in the previous experiment, D represents the analysis operator for a (shearlet) tight frame, constructed as in the previous example. The matrix K can be decomposed as $K = K_{\mathrm{p}} + K_{\mathrm{s}}$, where K_{p} represents a convolution operator that uses periodic boundary conditions. Therefore $K_{\mathrm{p}}^T K_{\mathrm{p}}$ and $D^T D = \alpha I$ are both diagonalized by the discrete Fourier basis matrix. The matrix K_{s} is a sparse matrix that modifies the values near the boundary to satisfy the replicate boundary conditions.

This problem has the canonical form (3.1) with

$$f(x) = 0, \qquad g(y_1, y_2) = \frac{1}{2}\|y_1 - b\|^2 + \gamma\|y_2\|_1, \qquad A = \begin{bmatrix} K \\ D \end{bmatrix}.$$

Since f is quadratic, the simplified mixed splitting algorithm of section 3.2.4 can be applied. However, as indicated at the end of section 3.2.4, additional simplifications are possible because g is a separable sum of a quadratic term and a non-quadratic term.

Define $g_1(y_1) = \|y_1 - b\|^2/2$ and $g_2(y_2) = \gamma\|y_2\|_1$. The primal-dual optimality conditions for (3.19) can be written as

$$0 \in \begin{bmatrix} 0 & K^T & D^T \\ -K & 0 & 0 \\ -D & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \nabla g_1^*(z_1) \\ \partial g_2^*(z_2) \end{bmatrix}.$$

(Note that $g_1^*(z_1) = \|z_1\|^2/2 + b^T z_1$ and $\partial g_1^*(z_1) = \{\nabla g_1^*(z_1)\} = \{z_1 + b\}$.) We use the splitting

$$\mathcal{A}(x, z) = \begin{bmatrix} 0 & K_{\mathrm{s}}^T & 0 \\ -K_{\mathrm{s}} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \partial g_2^*(z_2) \end{bmatrix}$$

and

$$\mathcal{B}(x, z) = \begin{bmatrix} 0 & K_{\mathrm{p}}^T & D^T \\ -K_{\mathrm{p}} & I & 0 \\ -D & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ b \\ 0 \end{bmatrix}.$$

Evaluating $(x, z_1, z_2) = (I + t\mathcal{A})^{-1}(\hat{x}, \hat{z}_1, \hat{z}_2)$ separates into two independent calculations: the solution of the linear equation

$$\begin{bmatrix} I & tK_{\mathrm{s}}^T \\ -tK_{\mathrm{s}} & I \end{bmatrix} \begin{bmatrix} x \\ z_1 \end{bmatrix} = \begin{bmatrix} \hat{x} \\ \hat{z}_1 \end{bmatrix}$$

and the evaluation of a prox-operator $z_2 = \mathrm{prox}_{tg_2^*}(\hat{z}_2)$. Evaluating the resolvent of $\mathcal{B}$ requires only solving a linear system. This system can be reduced to a system with a coefficient consisting of positive weighted sum of three terms: $I + \lambda K_{\mathrm{p}}^T K_{\mathrm{p}} + \nu D^T D$. Hence the resolvent of $\mathcal{B}$ can be evaluated efficiently via fast Fourier transforms. Notice that we removed the variable y which appeared in the method of section 3.2.4 for quadratic f. This reduces the size of the monotone inclusion problem significantly.

Figure 3.10 compares the performances of the Chambolle-Pock, ADMM, primal Douglas-Rachford, and primal-dual Douglas-Rachford algorithms. The image b is a degraded version of the "Barbara" image, resized to have size 256 by 256. The original image (scaled so that intensity values are between 0 and 1) was blurred with a 9 by 9 truncated Gaussian kernel with standard deviation $\sigma = 4$. Then Gaussian noise with zero mean and standard deviation 10^{-3} was added to each pixel of the blurred image. The parameter γ was chosen to give a visually appealing image reconstruction. A nearly optimal primal objective value $f^\star$ was computed by running the primal-dual Douglas-Rachford algorithm for $10,000$ iterations. In figure 3.10 the quantity $(f^k - f^\star)/f^\star$ is plotted against the iteration number k, where f^k is the primal objective value at iteration k. The original, blurry/noisy, and restored (by primal-dual DR) images are shown in figure 3.11.

For all methods, close to optimal fixed primal and dual step sizes (and overrelaxation parameters) were selected by trial and error. Primal and dual step sizes were implemented in the algorithms based on Douglas-Rachford by modifying g and A as in equation (3.2). The Chambolle-Pock step sizes s and t were chosen to satisfy $tsL^2 = 1$ where $L = (\|K_{\mathrm{p}}\|^2 + \|D\|^2)^{1/2} \approx \|A\|$. The norm of K_{p} can be computed analytically because $K_{\mathrm{p}}^T K_{\mathrm{p}}$ is diagonalized by the discrete Fourier basis. The norm of D is $\sqrt{\alpha}$ because $D^T D = \alpha I$.

The average elapsed time per iteration was 0.55 seconds for Chambolle-Pock, 0.78 seconds

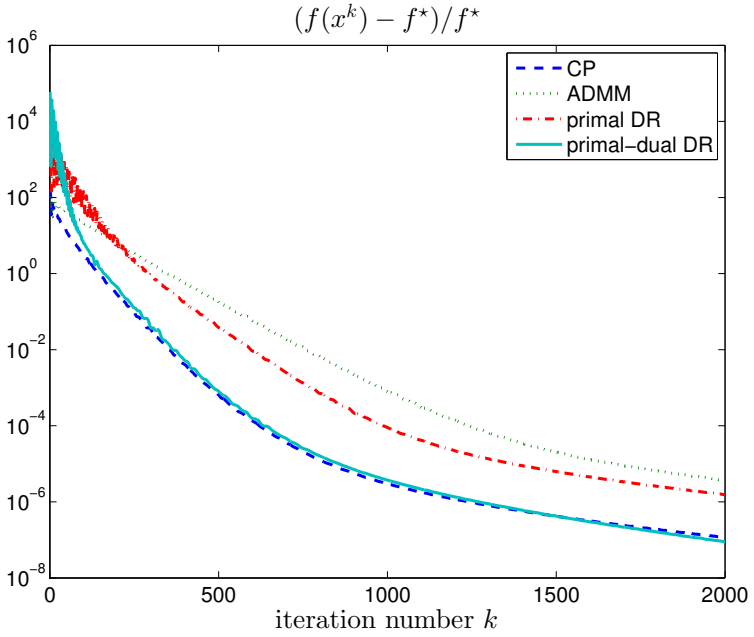


Figure 3.10: Relative optimality gap vs. iteration for the experiment in section 3.5.2.

(a) Original image. (b) Blurry, noisy image. (c) Restored image.

Figure 3.11: Result for the experiment in section 3.5.2.

for ADMM, 0.60 seconds for primal Douglas-Rachford, and 0.52 seconds for primal-dual Douglas-Rachford.

3.6 Primal-dual decomposition

The mixed splitting algorithm applies to problems in many areas beyond image reconstruction. For example, one often encounters optimization problems that are 'almost' separable: the functions f and g in (3.1) are block-separable and $A = B + C$ with B block-diagonal and C sparse. In this situation, evaluating the resolvents of f and g decomposes into independent subproblems which can be solved in parallel. A linear system with coefficient matrix $I + \lambda B^T B$ is also separable and can be solved by solving independent smaller equations. Systems with coefficient matrix $I + \lambda C^T C$ may be solved efficiently by exploiting sparsity. The mixed splitting algorithm therefore leads to primal-dual decomposition schemes for almost separable problems. This type of structure generalizes angular and dual-angular structure [Las70] found in problems that are separable except for a small number of coupling constraints ('dual decomposition') or coupling variables ('primal decomposition'); see also [CLC07, PC06]. The splitting methods discussed in this chapter can therefore be viewed as generalized primal-dual decomposition methods.

To illustrate this with an image deblurring application, we consider a spatially varying blurring model with a piecewise constant kernel. We assume the image can be partitioned into rectangular regions such that the blurring kernel is constant on each region (variations on this basic model are discussed in [NO98]). If pixels are ordered appropriately (one subimage after another), then the blurring operator can be represented by a matrix $K = K_{\mathrm{bd}} + K_{\mathrm{s}}$, where K_{bd} is block diagonal and K_{s} is sparse, and moreover each block of K_{bd} represents a spatially invariant convolution (applied to the corresponding subimage) using periodic boundary conditions. Linear systems involving the blocks of K_{bd} can be solved efficiently via the FFT. Similarly, a discrete gradient operator can be represented by a sum $D = D_{\mathrm{bd}} + D_{\mathrm{s}}$ of a block-diagonal and a sparse matrix, where each block of D_{bd} represents a spatially

invariant convolution (applied to a subimage), using periodic boundary conditions.

We consider again the deblurring model (3.16), where now the blurring kernel is only assumed to be piecewise constant (and replicate boundary conditions are used for the entire image). The matrix A in (3.17) can be decomposed as $A = B + C$ with

$$B = \begin{bmatrix} K_{\mathrm{bd}} \\ D_{\mathrm{bd}} \end{bmatrix}. \qquad C = \begin{bmatrix} K_{\mathrm{s}} \\ D_{\mathrm{s}} \end{bmatrix}. \tag{3.20}$$

Equations with coefficient $I+\lambda B^T B$ (for a given $\lambda > 0$) are easy to solve because $I+\lambda B^T B$ is block diagonal, with blocks that can be diagonalized by the DFT. Equations with coefficient $I + \lambda C^T C$ can be solved efficiently by exploiting sparsity. We solve (3.16) using the methods discussed in section 3.2.3, with the decomposition (3.20).

Figure 3.12 compares the performances of the Chambolle-Pock, ADMM, primal Douglas-Rachford, and primal-dual Douglas-Rachford algorithms. The noisy, blurry image b is a degraded version of the 256 by 256 "cameraman" image. The original image (scaled so that intensity values are between 0 and 1) was partitioned into four subimages, each of which was blurred with a 9 by 9 truncated Gaussian kernel. (The standard deviations were $4, 3.5, 3$, and 2 for the upper-left, upper-right,lower-left, and lower-right subimages, respectively.) Then salt and pepper noise was added to a random selection of 10% of the pixels. The parameter γ was chosen to give a visually appealing image reconstruction. A nearly optimal primal objective value $f^\star$ was computed by running the primal-dual Douglas-Rachford algorithm for $10,000$ iterations. In figure 3.12 the quantity $(f^k - f^\star)/f^\star$ is plotted against the iteration number k. (f^k is the primal objective value at iteration k.) The blurry/noisy and the restored (by primal-dual Douglas-Rachford) images are shown in figure 3.12. The figure also shows the image after five iterations of the primal-dual DR method. In this partially restored image the artifacts due to the spatially varying blurring operator are still visible.

For all methods, close to optimal fixed primal and dual step sizes (and overrelaxation parameters) were selected by trial and error. Primal and dual step sizes (for the Douglas-Rachford methods) were implemented by modifying g and A as in equation (3.2). The Chambolle-Pock step sizes s and t were chosen to satisfy $tsL^2 = 1$, where $L = (\|K_{\mathrm{bd}}\|^2 +$

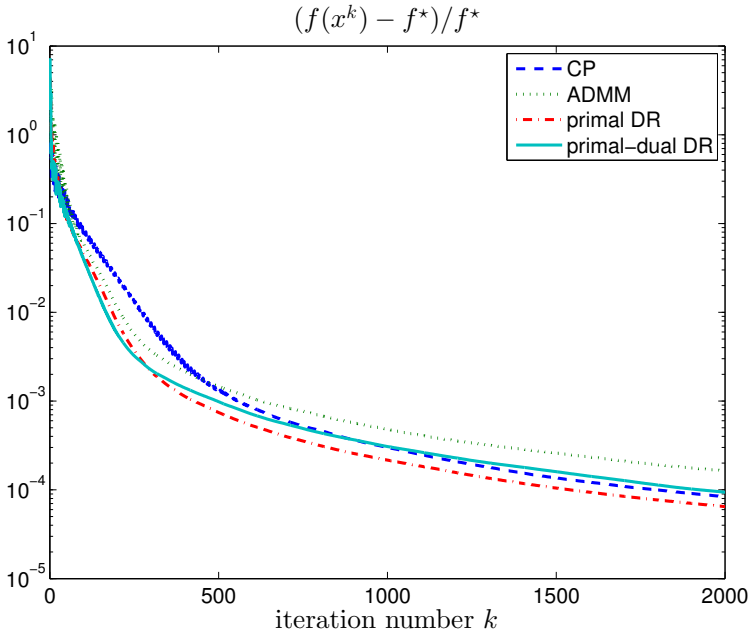


Figure 3.12: Relative optimality gap versus iteration number for the experiment in section 3.6.

(a) Blurry, noisy image. (b) After 5 iterations. (c) Restored image.

Figure 3.13: Result for the experiment in section 3.6.

	primal-dual	dual
PPA	proximal method of multipliers	method of multipliers
DR splitting	primal-dual DR splitting	ADMM

Table 3.1: Relation between various methods.

$8)^{1/2} \approx \|A\|$. (It can be shown that $\|D\|^2 \leq 8$.) The norm of K_{bd} can be computed analytically because the blocks of K_{bd} are diagonalized by the discrete Fourier basis. The average elapsed time per iteration was 0.086 seconds for Chambolle-Pock, 0.15 seconds for ADMM, and 0.12 seconds for both primal DR and primal-dual DR.

3.7 Conclusions

We have presented primal-dual operator splitting algorithms for solving convex optimization problems

$$\text{minimize} \quad f(x) + g(Ax)$$

where f and g are closed, convex functions with inexpensive proximal operators, and the matrix A can be split as a sum $A = B + C$ of two structured matrices. Our approach is to apply the Douglas-Rachford splitting method to the primal-dual optimality conditions (Karush-Kuhn-Tucker conditions), expressed as a monotone inclusion. The relationship of this approach to other well known methods is illustrated in table 3.1. The method of multipliers (or augmented Lagrangian method) [Roc76a] and the Bregman iteration [YOG08]) can be interpreted as solving the dual optimality condition

$$0 \in \partial g^*(z) - A\partial f^*(-A^T z) \tag{3.21}$$

using the proximal point algorithm (PPA). The ADMM and split-Bregman methods can be interpreted as solving the dual optimality condition using the Douglas-Rachford (DR) algorithm [Gab83]. The proximal method of multipliers [Roc76a] uses the proximal point algorithm to solve the primal-dual optimality conditions. We complete this picture by using the Douglas-Rachford method to solve the primal-dual optimality conditions. This method

can therefore be viewed as a primal-dual version of the ADMM or split-Bregman methods, or as an alternating direction version of the proximal method of multipliers. The primal-dual approach has the advantage that it lends itself to several interesting splitting strategies. In particular, the 'mixed splitting' strategy exploits additive structure $A = B + C$ in the linear term, where B and C have useful, but different types of matrix structure. We have illustrated this with applications in image deblurring, in which the matrix B can be diagonalized by a DFT and C is a sparse matrix.

In the numerical experiments we compared the primal-dual Douglas-Rachford splitting method with three other methods: the Douglas-Rachford method applied to a reformulated primal problem (also known as Spingarn's method), the ADMM algorithm (Douglas-Rachford applied to the dual), and the Chambolle-Pock algorithm. All methods depend on three algorithm parameters: a primal step size, a dual step size, and an overrelaxation parameter. Using different primal and dual step sizes can also be interpreted as first scaling the matrix A and g and then using a single step size. We observed that with a careful tuning of the algorithm parameters the four methods performed similarly. In all experiments, the primal-dual Douglas-Rachford approach is consistently one of the fastest methods. Compared with the primal and dual Douglas-Rachford approaches it has the advantage that it avoids the introduction of large numbers of auxiliary variables and constraints. An advantage over the Chambolle-Pock algorithm is that the step size selection does not require estimates or bounds on the norm of A. This is particularly important in large-scale applications where the norm of the linear operators is difficult to estimate (unlike the norms of the convolution operators encountered in image deblurring). As in ADMM and other applications of the Douglas-Rachford method, it is difficult to provide general guidelines about the choice of step sizes. When comparing algorithms we used constant step sizes, tuned by trial and error. More practical adaptive strategies for step size selection in the Douglas-Rachford method have been proposed in [HYW00, HLW03].

The approach to primal-dual operator splitting discussed in this chapter does not obvi-

ously generalize to the case where A is a sum of N structured matrices. A different approach, based on the consensus trick for monotone inclusion problems, will be discussed in chapter 5.

CHAPTER 4

Total variation image deblurring with space-varying kernel[1]

4.1 Introduction

In many popular approaches to non-blind image deblurring, a deblurred image is computed by solving an optimization problem of the form

$$\underset{x}{\text{minimize}} \quad \phi_{\text{f}}(Kx - b) + \phi_{\text{r}}(Dx) + \phi_{\text{c}}(x) \tag{4.1}$$

where $\phi_{\text{f}}, \phi_{\text{r}}$, and ϕ_{c} are convex penalty or indicator functions [HNO06, Vog02]. Here b is a vector containing the pixel intensities of an $M \times N$ blurry, noisy image, stored as a vector of length $n = MN$, for example, in column-major order. The optimization variable $x \in \mathbb{R}^n$ is the deblurred image. The matrix K models the blurring process, and is assumed to be known. The first term in (4.1) is often called the "data fidelity" term and encourages $Kx \approx b$. Typical choices for the penalty function ϕ_{f} in the data fidelity term include the squared L_2 norm, the L_1 norm, and the Huber penalty. The second term in the cost function is a regularization term. In total-variation (TV) deblurring [ROF92, RO94], the matrix D represents a discrete gradient operator, and ϕ_{r} is an L_1 norm or the isotropic norm

$$\left\| \begin{bmatrix} u \\ v \end{bmatrix} \right\|_{\text{iso}} = \sum_{i=1}^{n} \sqrt{u_i^2 + v_i^2}. \tag{4.2}$$

In tight frame regularized deblurring [Mal99, KSZ11], D represents the analysis operator for a tight frame and ϕ_{r} is the L_1 norm. The last term $\phi_{\text{c}}(x)$ can be added to enforce constraints on x such as $0 \leq x \leq 1$ by choosing ϕ_{c} to be the indicator function of the constraint set.

[1] **This chapter is based on the paper [OV].**

When devising efficient methods for solving (4.1), a key requirement is to exploit structure in K. Most work on image deblurring has modeled the blur as being spatially invariant, in which case K represents a convolution with a spatially invariant point spread function. This allows efficient multiplication by K and K^T by means of the fast Fourier transform. Additionally, linear systems involving K can be solved efficiently because K can be expressed as a product of the 2-dimensional DFT matrix, a diagonal matrix, and the inverse of the 2-dimensional DFT matrix. In combination with fast Fourier transform techniques, methods such as the Chambolle-Pock algorithm [CP11a], the Douglas-Rachford algorithm [LM79, CP07, OV14], and the alternating direction method of multipliers (ADMM) or split Bregman method [GO09, CTY13, AF13] are very effective for solving (4.1). By exploiting the space-invariant convolution structure, these methods achieve a very low per-iteration complexity, equal to the cost of a small number of FFTs.

In this chapter we address the problem of solving (4.1) under the assumption of *spatially variant* blur. We consider two models of spatially variant blur: the classical model of Nagy and O'Leary [NO98] and the related "Efficient Filter Flow" (EFF) model [HSS10]. The Nagy-O'Leary model expresses a spatially variant blur operator as a sum

$$K = \sum_{p=1}^{P} U_p K_p, \tag{4.3}$$

where each K_p represents a spatially invariant convolution operator, and the matrices U_p are nonnegative diagonal matrices that add up to the identity. The EFF model uses a model of the form

$$K = \sum_{p=1}^{P} K_p U_p, \tag{4.4}$$

with similar assumptions on K_p and U_p. Unlike the space-invariant convolution operator, the space-varying models (4.3) and (4.4) cannot be diagonalized using the DFT. The contribution of this chapter is to present Douglas-Rachford splitting methods for solving (4.1), for the two spatially variant blur models, with an efficiency that is comparable with FFT-based methods for spatially invariant blurring operators. Specifically, for many common types of fidelity and regularization terms, the cost per iteration is $O(PN^2 \log N)$ for N by N images.

Our approach is to use the structure of the blur operator to write deblurring problems in the canonical form

$$\underset{x}{\text{minimize}} \quad f(x) + g(Ax) \tag{4.5}$$

in such a way that the convex functions f and g have inexpensive proximal operators, and linear systems involving A can be solved efficiently. Problem (4.5) can then be solved by methods based on the Douglas-Rachford splitting algorithm, yielding algorithms whose costs are dominated by the calculation of order P fast Fourier transforms at each iteration. The key step, for both models, is the evaluation of the proximal operator of g. Our method for the Nagy-O'Leary model requires that ϕ_{f} be a separable sum of functions with inexpensive proximal operators; hence, it is able to handle most of the standard convex data fidelity terms, such as those using an L_1 or squared L_2 norm or the Huber penalty. For the Efficient Filter Flow model, our method requires that ϕ_{f} is the squared L_2 norm. Both methods can handle TV and tight frame regularization.

While much of the literature on image deblurring has assumed a spatially invariant blur, recent work has increasingly focused on restoring images degraded by spatially variant blur [NO98, BJN06, HSS10, HHS10, HSH11, Lev06, BB11, BBA12, BB12, JKZ10, DTS11, WSZ12]. Most of the effort has gone towards modeling and estimating spatially variant blur. In contrast, the problem of TV or tight frame regularized deblurring with a known spatially variant blur operator has not received much attention. In the original Nagy and O'Leary paper [NO98], non-blind deblurring is achieved by solving $Kx = b$ by the preconditioned conjugate gradient method, stopping the iteration early to mitigate the effects of noise. The same approach is taken in [BJN06], where the Nagy-O'Leary model is used in a blind deblurring algorithm. The paper [HSS10], which introduced the Efficient Filter Flow model, recovers x given K and b by finding a nonnegative least squares solution to $Kx = b$. (Note that [PC04] independently introduced a similar model.) A Bayesian framework with a hyper-Laplacian prior is used in [JKZ10] and [WSZ12], leading to non-convex optimization problems which are solved by iteratively re-weighted least squares or by adapting the method of [CL09]. [Lev06] performs the non-blind deblurring step using the Richardson-

Lucy algorithm. Most relevant to us, [BB11] uses the Nagy-O'Leary model and solves a TV deblurring problem using a method based on the domain decomposition approach given in [FLS10], and [BB12] takes a similar domain decomposition approach to TV deblurring using the Efficient Filter Flow model. However, [BB11] requires that ϕ_{f} be the squared L_2 norm, whereas our treatment of the Nagy-O'Leary model is able to handle other important data fidelity penalties such as the L_1 norm and the Huber penalty. Moreover, the papers [BB11, BB12] consider only total variation regularization, whereas our approach is able to handle tight frame regularization as well as total variation regularization. Our approach allows for constraints on x, enforced by the term $\phi_{\mathrm{c}}(x)$; such constraints are not considered in [BB11, BB12]. An additional difference between our approach and that of [BB11, BB12] is that at each iteration of the methods in [BB11, BB12], there are subproblems which must themselves be solved by an iterative method; thus [BB11, BB12] present multi-level iterative algorithms, with both inner and outer iterations. In our approach all subproblems at each iteration have simple closed form solutions.

This chapter is organized as follows. In section 4.2 we present the Nagy-O'Leary model of spatially variant blur and derive a method for solving (4.1) using this model, under the assumption that ϕ_{f} is separable. In section 4.3 we discuss the Efficient Filter Flow model of spatially variant blur, and derive a method for solving (4.1) using this model, in the case where ϕ_{f} is the squared L_2 norm. Section 4.4 demonstrates the effectiveness of these methods with some numerical experiments. The chapter concludes with a summary of main points in section 4.5.

4.2 The Nagy-O'Leary model

The Nagy-O'Leary model of spatially variant blur [NO98] assumes that the blur operator K has the form

$$K = \sum_{p=1}^{P} U_p K_p. \tag{4.6}$$

where

- each K_p, for $p = 1, \ldots, P$, performs a convolution with a space-invariant kernel,
- each U_p is diagonal with nonnegative diagonal entries,
- the matrices U_p sum to the identity matrix: $\sum_{p=1}^{P} U_p = I$.

Intuitively, U_p determines how much K_p contributes to each pixel in the blurry image.

We now present a method for solving problem (4.1) when K has the form (4.6). The method requires the special assumption that ϕ_{f} is a separable sum of functions with inexpensive proximal operators:

$$\phi_{\mathrm{f}}(x) = \sum_i \phi_{\mathrm{f}}^i(x_i).$$

This is the case in most applications, for example if ϕ_{f} is a squared L_2 norm ($\phi_{\mathrm{f}}^i(u) = u^2/2$), an L_1 norm ($\phi_{\mathrm{f}}^i(u) = |u|$), or the Huber penalty

$$\phi_{\mathrm{f}}^i(u) = \begin{cases} u^2/(2\eta) & |u| \leq \eta \\ |u| - \eta/2 & |u| \geq \eta, \end{cases}$$

where η is a positive parameter. We make no assumptions on ϕ_{r} and ϕ_{c}, except that they have inexpensive proximal operators. The method also requires that $D^T D$ is diagonalized by the discrete Fourier basis, which is true for both TV and tight frame regularization. Assuming K satisfies (4.6), we write problem (4.1) in the generic form (4.5) by defining

$$\begin{aligned} f(x) &= \phi_{\mathrm{c}}(x), \\ g(y_1, \ldots, y_P, w) &= \phi_{\mathrm{f}}(U_1 y_1 + \cdots + U_P y_P - b) + \phi_{\mathrm{r}}(w), \\ A &= \begin{bmatrix} K_1^T & \cdots & K_P^T & D^T \end{bmatrix}^T. \end{aligned}$$

We can apply the primal-dual splitting method of section 3.2.2 to this problem, once we first check that the prox-operators of f and g are inexpensive and that the linear system in step (3.10) can be solved efficiently.

We first examine the prox-operators of f and g. The proximal operator of f is the proximal operator of ϕ_{c}, which is assumed to be inexpensive. The function g is separable in

the variables $y = (y_1, \ldots, y_P)$ and w, and can be written as $g(y, w) = g_1(y) + g_2(w)$ where

$$g_1(y) = \phi_{\mathrm{f}}(U_1 y_1 + \cdots + U_P y_P - b), \qquad g_2(w) = \phi_{\mathrm{r}}(w).$$

Therefore the proximal operator of g at $(\hat{y}, \hat{w})$ is of the form $\mathrm{prox}_{tg}(\hat{y}, \hat{w}) = (\mathrm{prox}_{tg_1}(\hat{y}), \mathrm{prox}_{tg_2}(\hat{w}))$. The proximal operator of g_2 is the proximal operator of ϕ_{r}, which is assumed to be inexpensive.

The only non-obvious question is how to calculate the proximal operator of g_1. Let $U = \begin{bmatrix} U_1 & \cdots & U_P \end{bmatrix}$ and let $M = (UU^T)^{1/2}$. Then we can express g_1 as $g_1(y) = h(M^{-1}Uy - M^{-1}b)$, where

$$h(u) = \phi_{\mathrm{f}}(Mu) = \sum_i \phi_{\mathrm{f}}^i(M_{ii}u_i).$$

The invertibility of M follows from $\sum_{p=1}^P U_p = I$. Noting that $(M^{-1}U)(M^{-1}U)^T = I$, we can now use the composition rule (2.21) to express the proximal operator of g_1 in terms of the proximal operator of h. Moreover, the proximal operator of h can be evaluated efficiently using the separable sum rule (2.18) together with the rule (2.19). The resulting formula for the proximal operator of g_1 is

$$\mathrm{prox}_{tg_1}(\hat{y}) = (I - U^T M^{-2} U)\hat{y} + U^T M^{-2}(v + b), \tag{4.7}$$

where v is the vector whose ith component is

$$v_i = \mathrm{prox}_{tM_{ii}^2 \phi_{\mathrm{f}}^i}((U\hat{y} - b)_i). \tag{4.8}$$

Regarding the linear system in step (3.10), note that

$$A^T A = K_1^T K_1 + \cdots + K_P^T K_P + D^T D.$$

If D represents a discrete gradient operator using periodic boundary conditions, or if D represents a tight frame analysis operator (*i.e.*, $D^T D = \alpha I$) then $D^T D$ is diagonalized by the discrete Fourier basis. In this case, $A^T A$ is also diagonalized by the discrete Fourier basis, and linear systems with coefficient $I + \lambda A^T A$ can be solved efficiently via the FFT.

It's physically unrealistic to assume periodic boundary conditions when modeling the blurring process. On the other hand, we want the operators K_p to use periodic boundary conditions for computational efficiency. We can deal with this issue by using an "unknown boundary conditions" approach as in [AF13]. Recall that we're assuming ϕ_{f} is a separable sum:

$$\phi_{\mathrm{f}}(x) = \sum_i \phi_{\mathrm{f}}^i(x_i).$$

If pixel i is a border pixel (close enough to the edge of the image that boundary conditions are relevant), then we simply redefine the function ϕ_{f}^i to be identically zero. We then solve the same optimization problem as before. The fact that the operators K_p use periodic boundary conditions is no longer a problem. In this approach, it is as if we are "inpainting" the border of the recovered image x. Once we've computed x, we can then discard the inpainted border. (Discarding the border also removes any artifacts introduced by using periodic boundary conditions for D.)

4.3 The Efficient Filter Flow model

A popular variant of the Nagy-O'Leary model for spatially variant blur is the *Efficient filter flow* (EFF) model introduced in [HSS10]. This model is used in the recent papers [HHS10, HSH11, BBA12]. In the EFF model the blur operator K has the form

$$K = \sum_{p=1}^{P} K_p U_p \tag{4.9}$$

where the matrices K_p and U_p satisfy the same properties as in the Nagy-O'Leary model of section 4.2. In this section, we present a method for solving

$$\underset{x}{\text{minimize}} \quad \frac{1}{2}\|Kx - b\|^2 + \phi_{\mathrm{r}}(Dx) + \phi_{\mathrm{c}}(x) \tag{4.10}$$

in the case where K is given by (4.9). Problem (4.10) is the special case of problem (4.1) where $\phi_{\mathrm{f}}(x) = \frac{1}{2}\|x\|^2$. We assume, as before, that ϕ_{r} and ϕ_{c} are proper closed convex functions with inexpensive proximal operators. Additionally, we assume that D is a tight

frame operator ($D^T D = \alpha I$), but the method can also be used when $D^T D$ is very sparse (as in TV regularization).

Problem (4.10) can be expressed in the generic form (4.5) by defining

$$\begin{aligned} f(x) &= \phi_{\mathrm{c}}(x), \\ g(y_1, \ldots, y_P, w) &= \frac{1}{2}\| \sum_{p=1}^{P} K_p y_p - b\|_2^2 + \phi_{\mathrm{r}}(w), \\ A &= \begin{bmatrix} U_1^T & \cdots & U_P^T & D^T \end{bmatrix}^T. \end{aligned}$$

We can solve this problem using the primal-dual splitting method of section 3.2.2, but we first must check that the prox-operators of f and g are inexpensive and that the linear system of equations in step (3.10) can be solved efficiently.

Again we first examine the prox-operators of f and g. The proximal operator of f is the proximal operator of ϕ_{c}, which is assumed to be inexpensive. The function g is separable in the variables $y = (y_1, \ldots, y_P)$ and w, and can be written as $g(y, w) = g_1(y) + g_2(w)$ where

$$g_1(y) = \frac{1}{2}\|K_1 y_1 + \cdots + K_P y_P - b\|_2^2, \qquad g_2(w) = \phi_{\mathrm{r}}(w).$$

Therefore the proximal operator of g at $(\hat{y}, \hat{w})$ is of the form $\mathrm{prox}_{tg}(\hat{y}, \hat{w}) = (\mathrm{prox}_{tg_1}(\hat{y}), \mathrm{prox}_{tg_2}(\hat{w}))$. The proximal operator of g_2 is the proximal operator of ϕ_{r}, which is assumed to be inexpensive. The key point of this section is that g_1 also has an inexpensive proximal operator.

We can express g_1 as $g_1(y) = (1/2)\|By - b\|_2^2$, where $B = \begin{bmatrix} K_1 & \cdots & K_P \end{bmatrix}$. From rule (2.22) for the proximal operator of a quadratic function, the proximal operator of g_1 is given by

$$\mathrm{prox}_{tg_1}(\hat{y}) = (I - tB^T(I + tBB^T)^{-1}B)(\hat{y} + B^T b). \tag{4.11}$$

Because the matrix

$$I + tBB^T = I + t\sum_{p=1}^{P} K_p K_p^T$$

is diagonalized by the discrete Fourier basis, expression (4.11) for the proximal operator of g_1 can be evaluated efficiently using the FFT.

Regarding the linear system in step (3.10), note that the linear equation in each iteration has coefficient matrix

$$I + stA^TA = I + st(U_1^2 + \cdots + U_P^2 + D^TD).$$

This matrix is diagonal when D represents the analysis operator of a tight frame, and very sparse when D represents a discrete gradient operator.

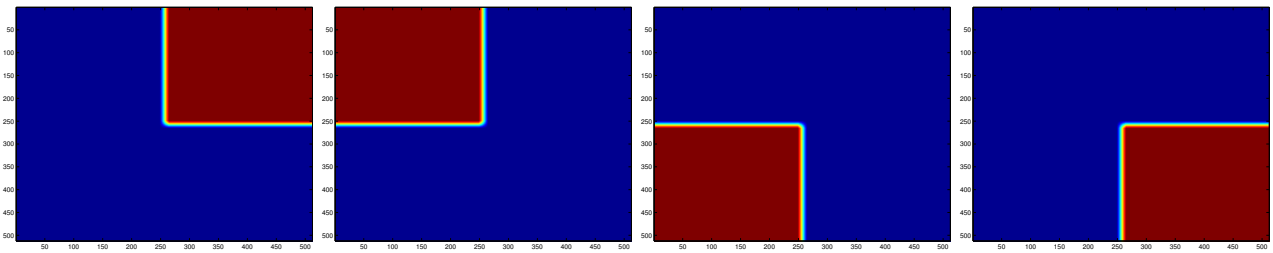
4.4 Experiments

All experiments were performed on a computer with a 3.70 GHz Intel Xeon(R) E5-1620 v2 processor with 8 cores and 7.7 GB of RAM. The code was written in MATLAB using MATLAB version 8.1.0.604 (R2013a).

4.4.1 Spatially variant Gaussian blur

We demonstrate the method of section 4.2 by restoring an image which has been blurred by an operator K that is described by the Nagy-O'Leary model (4.6). The image used is the 512 by 512 "Barbara" image, scaled to have intensity values between 0 and 1. We set $P = 4$, corresponding to the four quadrants of the image. For $p = 1, \ldots, 4$, the matrix K_p performs a convolution with a 17 by 17 truncated Gaussian kernel, with standard deviation $\sigma_p = p$ pixels. The matrix U_p zeros out components away from the pth quadrant, as visualized in figure 4.1 (where red corresponds to a value of 1 and blue corresponds to a value of 0). When creating the blurry image, "replicate" boundary conditions were used – meaning that an intensity value at a location outside the image is assumed to be equal to the intensity at the nearest pixel in the image. Gaussian noise with zero mean and standard deviation 10^{-3} was added to each pixel of the blurred image. After Gaussian noise was added to each pixel, 10% of the pixels (chosen at random) were corrupted by "salt and pepper" noise. (The intensity value at each corrupted pixel was randomly set to either 0 or 1.)

We handle boundary conditions as described in section 4.2, using the "unknown boundary



(a) U_1. (b) U_2 (c) U_3 (d) U_4

Figure 4.1: Visualization of the matrices U_p in section 4.4.1. Red corresponds to a value of 1, and blue corresponds to a value of 0. The numerical values transition smoothly from 0 to 1.

conditions" approach. First the blurry image is zero padded to have size 528 by 528, then a restored image is computed by solving

$$\underset{x}{\text{minimize}} \quad \phi_{\text{f}}(Kx - b) + \gamma \|Dx\|_{\text{iso}}. \tag{4.12}$$

We take $\phi_{\text{f}}(x) = \sum_i \phi_{\text{f}}^i(x_i)$, where ϕ_{f}^i is the Huber penalty (with parameter $\eta = 10^{-3}$) if pixel i is not a border pixel, and ϕ_{f}^i is identically zero if pixel i is a border pixel. The optimization variable is $x \in \mathbb{R}^{528^2}$, and D is the matrix representation of a discrete gradient operator $\tilde{D} : \mathbb{R}^{528\times 528} \to \mathbb{R}^{528\times 528\times 2}$, defined by:

$$(\tilde{D}x)_{i,j,1} = x_{i+1,j} - x_{ij}, \qquad (\tilde{D}x)_{i,j,2} = x_{i,j+1} - x_{ij}.$$

($\tilde{D}$ uses periodic boundary conditions.) Once a deblurred image is computed by solving (4.12), the inpainted boundary is discarded, yielding a restored image of size 512 by 512. The parameter γ is chosen to give a visually appealing image reconstruction. The original, blurry, and restored images are shown in figure 4.2. Notice that the lower right quadrant is blurred the most. It's interesting to look closely and see that more detail is recovered in the upper right quadrant, where the blurring was less severe.

In figure 4.3a, the quantity $\|x^k - x^\star\|/\|x^\star\|$ is plotted against the iteration number k. Here x^k is the estimate of the solution to (4.12) at iteration k, and $x^\star$ is a nearly optimal primal variable which was computed by running the method of section 4.2 for $10,000$ iterations. In figure 4.3b, the quantity $(F^k - F^\star)/F^\star$ is plotted against the iteration number k. Here

(a) Original image. (b) Blurry, noisy image. (c) Restored image.

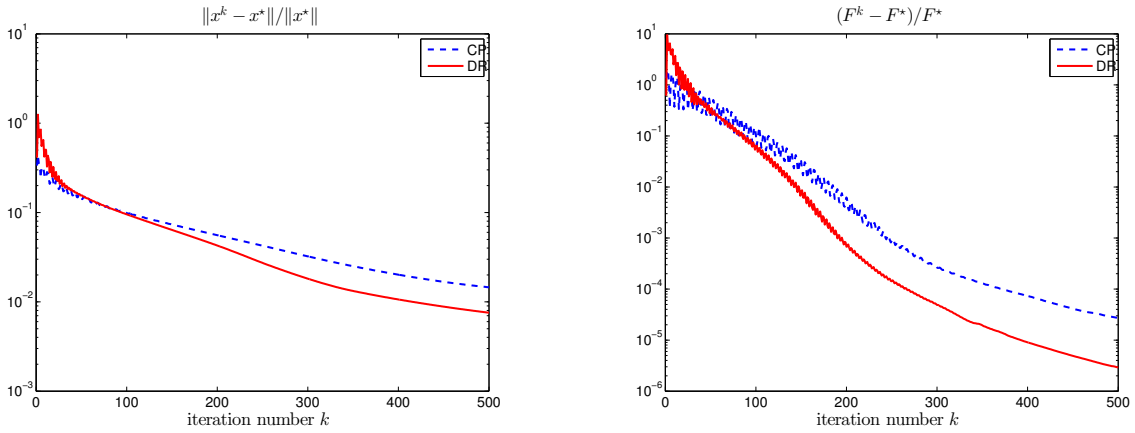
Figure 4.2: Result for the experiment in section 4.4.1.

F^k is the primal objective function value at iteration k, and $F^\star$ is a nearly optimal primal objective function value, which was computed by running the method of section 4.2 for $10,000$ iterations. To illustrate the convergence properties of the algorithm, we show the error for the first 500 iterations. However we observed that after about 180 iterations the estimate x^k was visually indistinguishable from $x^\star$.

Figure 4.3 compares the performance of the method of section 4.2 (labeled "DR") with the performance of the well known Chambolle-Pock method [CP11a] (labeled "CP"). Chambolle-Pock minimizes $f(x) + g(Ax)$, where f and g are proper closed convex functions, via the iteration

$$\begin{aligned}
\bar{x}^k &= \mathrm{prox}_{tf}(x^{k-1} - tA^T z^{k-1}), \\
\bar{z}^k &= \mathrm{prox}_{sg^*}(z^{k-1} + sA(2\bar{x}^k - x^{k-1})), \\
(x^k, z^k) &= \rho(\bar{x}^k, \bar{z}^k) + (1-\rho)(x^{k-1}, z^{k-1}).
\end{aligned}$$

Here s and t are step sizes, and $\rho \in (0,2)$ is an overrelaxation parameter. (This overrelaxed version of Chambolle-Pock is presented in [Con13].) The step sizes s and t are required by convergence proofs to satisfy $st\|A\|^2 \leq 1$ [Con13], and we choose them so that $st\|A\|^2 = 1$. We precompute $\|A\|$ using power iteration. When solving (4.12) by Chambolle-Pock, we



(a) Relative error vs. iteration.

(b) Relative optimality gap vs. iteration.

Figure 4.3: Relative error vs. iteration and relative optimality gap vs. iteration for the experiment in section 4.4.1. The solid line shows the convergence of the primal-dual Douglas-Rachford method, and the dashed line shows the convergence of the Chambolle-Pock method.

take

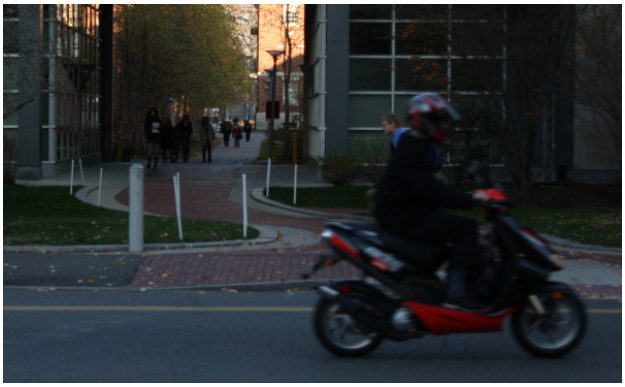
$$A = \begin{bmatrix} K \\ D \end{bmatrix}$$

and

$$f(x) = 0, \qquad g(y_1, y_2) = \phi_{\mathrm{f}}(y_1 - b) + \gamma \|y_2\|_{\mathrm{iso}}$$

for all x, y_1, y_2. One of the main advantages of Chambolle-Pock is that it requires only *applying* A and A^T (at each iteration), and never requires solving a linear system involving A. It is therefore very well suited to exploit the structure in the Nagy-O'Leary model.

As shown in figure 4.3, in this example the method of section 4.2 converges in fewer iterations than the Chambolle-Pock method. Moreover, the time per iteration is nearly the same for both methods: 0.19 seconds for Chambolle-Pock compared to 0.20 seconds for the method of section 4.2. In this experiment, close to optimal fixed step sizes and overrelaxation parameters for both methods were chosen by trial and error. (In particular, the Chambolle-Pock step sizes were not taken to be equal.)



(a) Blurry image.

(b) Restored image.

Figure 4.4: Result for the experiment in section 4.4.2.

Figure 4.5: The segmentation computed using the code from [CZF10].

4.4.2 Motion deblurring

In this section we combine our approach to spatially variant deblurring with the motion segmentation algorithm of [CZF10]. The algorithm presented in [CZF10] segments out a motion-blurred region from an otherwise sharp input image and estimates a motion blur kernel for this region, but does not compute a deblurred image.

A blurry image (of size 367 by 600) is shown in figure 4.4a, and a segmentation of this image computed using the code provided by [CZF10] is shown in figure 4.5. The algorithm of [CZF10] estimated a horizontal motion of 6 pixels during the time the camera shutter was open, but a slightly better image reconstruction was obtained using the motion blur kernel

$(1/7)\begin{bmatrix}1 & 1 & 1 & 1 & 1 & 1 & 1\end{bmatrix}$, which corresponds to a horizontal motion of 7 pixels.

We use the Nagy-O'Leary model (4.6) with $P = 2$. The matrix K_1 performs a convolution with the 7 pixel motion blur kernel given above, and K_2 is the identity matrix (we assume negligible blur for the background region). The matrix U_1 zeros out background pixels without altering foreground intensity values, while the matrix U_2 zeros out foreground pixels without altering background intensity values. We first zero pad the blurry image to have size 373 by 606, then compute a deblurred image by solving (4.12) using the method of section 4.2. (We deblur each color channel of the blurry image separately, to obtain a restored color image.) We use a quadratic data fidelity function ϕ_{f}. The discrete gradient operator D is defined as in section 4.4.1. The parameter γ is chosen to give a visually appealing image reconstruction. Once a deblurred image is computed by solving (4.12), the inpainted boundary is discarded, yielding a restored image of size 367 by 600. (Because K_2 is the identity, this effort to handle boundary conditions correctly is actually unnecessary in this example.)

When solving (4.12), we ran the primal-dual Douglas-Rachford method discussed in section 3.2.2 for 250 iterations. The step sizes and overrelaxation parameter were set to the same values as in section 4.4.1. The total run time was 79 seconds to compute a segmentation and estimate a motion blur kernel (using the code from [CZF10]), and then 164 seconds to compute a restored color image.

The deblurred image is shown in figure 4.4b. When we zoom in, the letters on the motorcycle in the restored image are nearly legible now, and appear to say "racing".

4.5 Conclusion

Beginning with [ROF92] over 20 years ago, a substantial literature has been devoted to non-blind TV image deblurring under the assumption of spatially invariant blur. In this chapter, we have extended this line of research by presenting efficient methods for TV or tight frame regularized deblurring using two fundamental models of *spatially variant* blur:

the classical Nagy-O'Leary model and the related Efficient Filter Flow model. In the case of the Nagy-O'Leary model, our method requires that the data fidelity function ϕ_{f} in problem (4.1) is a separable sum of functions with inexpensive proximal operators. This includes most standard data fidelity functions such as the squared L_2 norm, the L_1 norm, and the Huber penalty. In the case of the Efficient Filter Flow model, our method requires that ϕ_{f} is the squared L_2 norm. Both methods can handle TV and tight frame regularization, as well as constraints such as box constraints on the recovered image x.

The Nagy-O'Leary model and the Efficient Filter Flow model both express a spatially variant blur operator as a sum of P terms, each term involving a spatially invariant blur operator. For the non-blind deblurring algorithms presented in this chapter, the computational cost is dominated by the cost of computing order P fast Fourier transforms at each iteration. The cost per iteration is $O(PN^2 \log N)$ for N by N images. Thus, we can solve (4.1), for the two spatially variant blur models, with an efficiency that is comparable with the efficiency of FFT-based methods that assume a spatially invariant blur model. In the case where $P = 1$, the methods presented in this chapter reduce to state of the art proximal algorithms for spatially invariant TV or tight frame regularized deblurring (see [OV14] for example).

The experiment in section 4.4.2 gives an example where the methods of this chapter can be combined with a motion segmentation algorithm to obtain a blind motion deblurring algorithm. In future work, it would be interesting to explore combining the non-blind algorithms presented here with successful approaches to blind deblurring, such as recent algorithms that restore images degraded by camera shake. Along these lines, the paper [HSH11] presents a blind deblurring algorithm that combines the structural constraints of a Projective Motion Path Blur (PMPB) model with the efficiency of the Efficient Filter Flow model, reaping the benefits of both frameworks. This demonstrates that PMPB models, a subject of much current interest, can be usefully combined with classical models of spatially variant blur, where our methods are applicable. Another goal would be to develop proximal algorithms that work with PMPB models directly.

Many blind deblurring algorithms, including [HSH11], use a non-convex regularizer based on the statistics of natural images, rather than using TV regularization. Another future research direction is to adapt the proximal algorithms given in this chapter, to handle these non-convex regularizers.

CHAPTER 5

Consensus Douglas-Rachford for monotone inclusion problems

5.1 Consensus Douglas-Rachford

In this chapter we discuss two "consensus" versions of the Douglas-Rachford method, which are able to solve monotone inclusion problems that involve a sum of more than two monotone operators. These methods give us a new method for handling additive structure in A for the canonical problem (1.1): when A is a sum of N structured matrices, the primal-dual optimality conditions (2.11) involve a sum of $N + 1$ monotone operators, and consensus Douglas-Rachford methods can be applied. (And there is no difficulty when $N > 2$, a case which can't obviously be handled by the primal-dual operator splitting approach of chapter 3.) In section 5.3 we revisit an experiment from chapter 3, and show that we can improve on the earlier results.

The consensus Douglas-Rachford method for minimizing a sum of N closed convex functions [CP07, BC11a] is seen to be a special case of these methods. Because ADMM can be interpreted as Douglas-Rachford applied to the dual problem, the consensus Douglas-Rachford methods lead to N-operator "consensus" versions of ADMM.

5.1.1 Derivation based on consensus trick

Let $f_0, \ldots, f_{N-1}$ be proper closed convex functions. The consensus Douglas-Rachford method for convex optimization [CP07, BC11a] solves

$$\text{minimize} \quad \sum_{i=0}^{N-1} f_i(u) \tag{5.1}$$

by applying Douglas-Rachford to the equivalent problem

$$\text{minimize} \quad \underbrace{\sum_{i=0}^{N-1} f_i(x_i)}_{g(x)} + \underbrace{\delta_C(x_0, \ldots, x_{N-1})}_{h(x)},$$

where the optimization variable is $x = (x_0, \ldots, x_{N-1})$ and δ_C is the indicator function of

$$C = \{(x_0, \ldots, x_{N-1}) \mid x_0 = \cdots = x_{N-1}\}.$$

Another method for solving (5.1) can be derived by applying Douglas-Rachford to the equivalent problem

$$\text{minimize} \quad \underbrace{\sum_{i=1}^{N-1} f_i(x_i)}_{g(x)} + \underbrace{f_0(x_1) + \delta_C(x_1, \ldots, x_{N-1})}_{h(x)}$$

where the optimization variable is now $x = (x_1, \ldots, x_{N-1})$. In this approach the reformulated problem has fewer variables, but the prox-operator of f_0 is not computed in parallel with the prox-operators of $f_i, i = 1, \ldots, N-1$. The resulting algorithm is a true extension of the Douglas-Rachford method, in the sense that it reduces to the Douglas-Rachford method in the case where $N = 2$.

Similar tricks can be applied to the monotone inclusion problem

$$0 \in \sum_{i=0}^{N-1} F_i(u) \tag{5.2}$$

where each F_i is a maximal monotone operator. A consensus Douglas-Rachford method for the monotone inclusion problem (5.2) can be derived by applying Douglas-Rachford to the

equivalent problem

$$0 \in \underbrace{\begin{bmatrix} F_0(x_0) \\ \vdots \\ F_{N-1}(x_{N-1}) \end{bmatrix}}_{G(x)} + \underbrace{\partial\delta_C(x_0, \ldots, x_{N-1})}_{H(x)} \tag{5.3}$$

with variable $x = (x_0, \ldots, x_{N-1})$. The equivalence of (5.2) and (5.3) follows from the fact that

$$\partial\delta_C(x_0, \ldots, x_{N-1}) = \begin{cases} C^\perp = \left\{(g_0, \ldots, g_{N-1}) \mid \sum_{i=0}^{N-1} g_i = 0\right\} & \text{if } x_0 = \cdots = x_{N-1}, \\ \emptyset \quad \text{otherwise.} \end{cases}$$

We will refer to the resulting algorithm as the standard consensus Douglas-Rachford method, or just as the consensus Douglas-Rachford method.

A different (and in some cases advantageous) algorithm for solving problem (5.2) can be derived by applying Douglas-Rachford to the equivalent problem

$$0 \in \underbrace{\begin{bmatrix} F_1(x_1) \\ \vdots \\ F_{N-1}(x_{N-1}) \end{bmatrix}}_{G(x)} + \underbrace{\begin{bmatrix} F_0(x_1) \\ 0 \\ \vdots \\ 0 \end{bmatrix} + \partial\delta_C(x_1, \ldots, x_{N-1})}_{H(x)} \tag{5.4}$$

where the variable is now $x = (x_1, \ldots, x_{N-1})$. In this version, the reformulated problem has fewer variables, but the resolvent of F_0 is not computed in parallel with the resolvents of $F_i, i = 1, \ldots, N-1$. We'll refer to the resulting algorithm as "extended Douglas-Rachford" to distinguish it from the consensus Douglas-Rachford algorithm derived above (using equation (5.3)), and also because it reduces to the Douglas-Rachford method in the case $N = 2$.

To evaluate the resolvent of H, let $u = (I + tH)^{-1}(x)$. Then

$$\begin{bmatrix} x_1 \\ \vdots \\ x_{N-1} \end{bmatrix} \in \begin{bmatrix} u_1 + tF_0(u_1) \\ u_2 \\ \vdots \\ u_{N-1} \end{bmatrix} + \partial\delta_C(u_1, \ldots, u_{N-1}). \tag{5.5}$$

This implies that $u_1 = \cdots = u_{N-1}$ (otherwise $\partial\delta_C(u_1, \ldots, u_{N-1})$ would be empty). Note that $(g_1, \ldots, g_{N-1}) \in \partial\delta_C(u_1, \ldots, u_1) \iff g_1 + \cdots + g_{N-1} = 0$. Summing all components on each side of (5.5), we obtain

$$\sum_{i=1}^{N-1} x_i = ((N-1)I + tF_0)(u_1)$$

which implies that

$$\begin{aligned} u_1 &= ((N-1)I + tF_0)^{-1} \left(\sum_{i=1}^{N-1} x_i \right) \\ &= \left(I + \frac{t}{N-1} F_0 \right)^{-1} \left(\frac{1}{N-1} \sum_{i=1}^{N-1} x_i \right). \end{aligned}$$

Note that when $F_0 = 0$, the extended Douglas-Rachford algorithm reduces to the consensus Douglas-Rachford algorithm for solving $0 \in \sum_{i=1}^{N-1} F_i(x)$. Hence, the consensus Douglas-Rachford algorithm is a special case of the extended Douglas-Rachford algorithm.

5.1.2 Alternate derivation

As mentioned in section 5.1.1, the extended Douglas-Rachford method reduces to the Douglas-Rachford method in the case where $N = 2$. It's interesting to note that the extended Douglas-Rachford method can also be derived by directly generalizing the derivation of the Douglas-Rachford method which was presented in section 2.4.

Our goal is to solve the problem

$$0 \in \sum_{i=0}^{N-1} F_i(x)$$

where each F_i is maximal monotone. Let $t > 0$. Then

$$0 \in \sum_{i=0}^{N-1} F_i(x) \quad \iff \quad 2(N-2)x \in ((N-1)I + tF_0)(x) + \sum_{i=1}^{N-1} (I + tF_i)(x). \tag{5.6}$$

Introducing new variables $z_i \in (I + tF_i)(x)$ for $i = 1, \ldots, N-1$, we see that the inclusion

(5.6) is equivalent to

$$\begin{aligned}
x &= (I + tF_i)^{-1}(z_i) \quad \text{for } i = 1, \ldots, N-1 \\
x &= ((N-1)I + tF_0)^{-1} \left(\sum_{i=1}^{N-1} 2x - z_i \right) \\
&= \left(I + \frac{t}{N-1} F_0 \right)^{-1} \left(\frac{1}{N-1} \sum_{i=1}^{N-1} 2x - z_i \right) \\
&= R_{F_0} \left(\frac{1}{N-1} \sum_{i=1}^{N-1} \mathrm{ref}_{F_i}(z_i) \right).
\end{aligned}$$

(We are using the notation $R_{F_0} = \left(I + \frac{t}{N-1}F_0\right)^{-1}$, $R_{F_i} = (I + tF_i)^{-1}$ for $i = 1, \ldots, N-1$, and $\mathrm{ref}_{F_i}(z_i) = 2R_{F_i}(z_i) - z_i$ for $i = 1, \ldots, N-1$.) Combining these equations, we see that

$$\begin{aligned}
& R_{F_i}(z_i) = R_{F_0} \left(\frac{1}{N-1} \sum_{i=1}^{N-1} \mathrm{ref}_{F_i}(z_i) \right) \\
\implies & 2R_{F_i}(z_i) - \mathrm{ref}_{F_i}(z_i) = 2R_{F_0} \left(\frac{1}{N-1} \sum_{i=1}^{N-1} \mathrm{ref}_{F_i}(z_i) \right) - \mathrm{ref}_{F_i}(z_i) \\
\implies & z_i = 2R_{F_0} \left(\frac{1}{N-1} \sum_{i=1}^{N-1} \mathrm{ref}_{F_i}(z_i) \right) - \mathrm{ref}_{F_i}(z_i)
\end{aligned}$$

for $i = 1, \ldots, N-1$. These equations together give us a fixed point equation for $z = (z_1, \ldots, z_{N-1})$:

$$z = S(T(z))$$

where

$$T(z_1, \ldots, z_{N-1}) = (\mathrm{ref}_{F_1}(z_1), \ldots, \mathrm{ref}_{F_{N-1}}(z_{N-1})) \tag{5.7}$$

and

$$S(u_1, \ldots, u_{N-1}) = (2R_{F_0}(\bar{u}) - u_1, \ldots, 2R_{F_0}(\bar{u}) - u_{N-1})$$

where

$$\bar{u} = \frac{1}{N-1} \sum_{i=1}^{N-1} u_i.$$

The operators S and T can be viewed as "generalized" reflection operators, and we will show that (like ordinary reflection operators) S and T are nonexpansive. It follows that we can

solve for z via the fixed point iteration

$$z^+ = \frac{1}{2}z + \frac{1}{2}S(T(z)).$$

By introducing some intermediate variables, this iteration can be written as

$$\begin{aligned} x_i^k &= (I + tF_i)^{-1}(z_i^{k-1}) \quad \text{for } i = 1, \ldots, N-1 \\ y^k &= \left(I + \frac{t}{N-1}F_0\right)^{-1} \left(2\bar{x}^k - \bar{z}^{k-1}\right) \\ z_i^k &= z_i^{k-1} + y^k - x_i^k \quad \text{for } i = 1, \ldots, N-1. \end{aligned}$$

(We use the notation $\bar{x} = \frac{1}{N-1}\sum_{i=1}^{N-1} x_i$ and $\bar{z} = \frac{1}{N-1}\sum_{i=1}^{N-1} z_i$.) This iteration is the same as the iteration for the extended Douglas-Rachford algorithm derived in section 5.1.1.

5.1.3 Proof that S and T are nonexpansive

The nonexpansiveness of T follows immediately from equation (5.7) and the nonexpansiveness of the reflection operators ref_{F_i}. Moreover, showing that S is nonexpansive is no more difficult than showing that a reflection operator is nonexpansive. We give the details here.

The ith block of $S(u_1, \ldots, u_{N-1}) - S(v_1, \ldots, v_{N-1})$ is

$$B_i = 2R_{F_0}(\bar{u}) - 2R_{F_0}(\bar{v}) - (u_i - v_i)$$

where

$$\bar{u} = \frac{1}{N-1}\sum_{i=1}^{N-1} u_i \quad \text{and } \bar{v} = \frac{1}{N-1}\sum_{i=1}^{N-1} v_i.$$

Note that

$$\|B_i\|^2 = 4\|R_{F_0}(\bar{u}) - R_{F_0}(\bar{v})\|^2 - 4\left\langle R_{F_0}(\bar{u}) - R_{F_0}(\bar{v}), u_i - v_i \right\rangle + \|u_i - v_i\|^2.$$

Hence

$$\begin{aligned}
& \|S(u_1,\dots,u_{N-1}) - S(v_1,\dots,v_{N-1})\|^2 \\
= \quad & \sum \|B_i\|^2 \\
= \quad & \sum 4\|R_{F_0}(\bar{u}) - R_{F_0}(\bar{v})\|^2 - 4\left\langle R_{F_0}(\bar{u}) - R_{F_0}(\bar{v}), \sum u_i - \sum v_i \right\rangle + \sum \|u_i - v_i\|^2 \\
= \quad & 4(N-1)\|R_{F_0}(\bar{u}) - R_{F_0}(\bar{v})\|^2 - 4(N-1)\left\langle R_{F_0}(\bar{u}) - R_{F_0}(\bar{v}), \bar{u} - \bar{v} \right\rangle + \sum \|u_i - v_i\|^2 \\
\leq \quad & \sum \|u_i - v_i\|^2.
\end{aligned}$$

This shows that S is nonexpansive. In the last step we used the inequality

$$\|R_{F_0}(a) - R_{F_0}(b)\|^2 \leq \langle R_{F_0}(a) - R_{F_0}(b), a - b\rangle$$

which expresses the fact that R_{F_0} is firmly nonexpansive.

5.1.4 Equivalent form of extended Douglas-Rachford

We now derive an equivalent form of the extended Douglas-Rachford algorithm, which will be useful when deriving an extended version of ADMM. By starting with the y update, the extended Douglas-Rachford iteration can be written as

$$\begin{aligned}
y^k &= \left(I + \frac{t}{N-1}F_0\right)^{-1}\left(\frac{1}{N-1}\sum_{i=1}^{N-1} 2x_i^{k-1} - z_i^{k-1}\right) \\
z_i^k &= z_i^{k-1} + y^k - x_i^{k-1} \quad \text{for } i = 1,\dots,N-1 \\
x_i^k &= (I + tF_i)^{-1}(z_i^{k-1} + y^k - x_i^{k-1}) \quad \text{for } i = 1,\dots,N-1.
\end{aligned}$$

The way this is written, we can put the z_i update at the end. Making a change of variable $w_i = z_i - x_i$, the iteration becomes

$$\begin{aligned}
y^k &= \left(I + \frac{t}{N-1}F_0\right)^{-1}(\bar{x}^{k-1} - \bar{w}^{k-1}) \\
x_i^k &= (I + tF_i)^{-1}(y^k + w_i^{k-1}) \quad \text{for } i = 1,\dots,N-1 \\
w_i^k &= w_i^{k-1} + y^k - x_i^k \quad \text{for } i = 1,\dots,N-1.
\end{aligned}$$

This iteration can be compared with the equivalent version of the Douglas-Rachford iteration given in equations (2.28) – (2.30).

5.1.5 Primal-dual operator splitting revisited

In chapter 3, we presented a primal-dual operator splitting approach to solving the canonical problem (1.1). However, the approach of chapter 3 doesn't obviously extend to the case where A is a sum of N operators, with $N > 2$.

A different way to handle additive structure in A is to use a consensus Douglas-Rachford method. In the case where $A = \sum_{i=1}^{N} A_i$, the primal-dual optimality conditions (2.11) can be written as

$$0 \in \begin{bmatrix} \partial f(x) \\ \partial g^*(z) \end{bmatrix} + \sum_{i=1}^{N} \begin{bmatrix} 0 & A_i^T \\ -A_i & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix}. \tag{5.8}$$

The monotone operator on the right is a sum of $N+1$ "easy" monotone operators (assuming each A_i has structure we can exploit). Thus this monotone inclusion problem can be solved by either the consensus Douglas-Rachford method or the extended Douglas-Rachford method. In section 5.3, we'll see an experiment where this approach compares favorably with the methods of chapter 3.

5.2 Extensions of ADMM

Because ADMM can be interpreted as solving the dual problem with Douglas-Rachford, the extended Douglas-Rachford algorithm can be used to derive an extended version of ADMM. A different consensus version of ADMM can be derived by applying the standard consensus Douglas-Rachford method to the dual problem.

Let $f_0, f_1, \ldots, f_{N-1}$ be proper, closed, convex functions. We'll consider the problem

$$\begin{aligned} \text{minimize} \quad & \sum_{i=0}^{n-1} f_i(x_i) \\ \text{subject to} \quad & \sum_{i=0}^{N-1} A_i x_i = b \end{aligned} \tag{5.9}$$

with variables $x_i, i = 0, \ldots, N-1$. Each A_i is a linear transformation (or matrix) and b is a vector. The dual problem (expressed as a minimization problem) is

$$\text{minimize} \quad \langle z, b \rangle + \sum_{i=0}^{N-1} f_i^*(-A_i^T z) \tag{5.10}$$

with variable z. The dual problem has the form

$$\text{minimize} \quad \sum_{i=0}^{N-1} F_i(z)$$

where $F_0(z) = \langle z, b \rangle + f_0^*(-A_0^T z)$ and $F_i(z) = f_i^*(-A_i^T z)$ for $i = 1, \ldots, N-1$. We'll solve (5.10) by the extended Douglas-Rachford method. Intermediate steps will require us to evaluate the prox-operator of each function F_i, which is achieved using equations (2.39) – (2.40).

5.2.1 Derivation of ADMM iteration

Because the variable we're solving for is called z, we'll rewrite the extended Douglas-Rachford iteration using different letters:

$$\begin{aligned} p^k &= \left(I + \frac{t}{N-1} F_0\right)^{-1} (\bar{z}^{k-1} - \bar{w}^{k-1}) \\ z_i^k &= (I + tF_i)^{-1}(p^k + w_i^{k-1}) \quad \text{for } i = 1, \ldots, N-1 \\ w_i^k &= w_i^{k-1} + p^k - z_i^k \quad \text{for } i = 1, \ldots, N-1. \end{aligned}$$

Using (2.39)–(2.40) to evaluate prox-operators, the p update becomes

$$x_0^k = \underset{u}{\operatorname{argmin}} \quad f_0(u) + \frac{t}{2(N-1)} \left\| A_0 u - b + \sum_{j=1}^{N-1} \frac{(z_j^{k-1} - w_j^{k-1})}{t} \right\|^2$$
$$p^k = \bar{z}^{k-1} - \bar{w}^{k-1} + \frac{t}{N-1}(A_0 x_0^k - b)$$

and the z and w updates become

$$x_i^k = \underset{u}{\operatorname{argmin}} \quad f_i(u) + \frac{t}{2} \left\| A_i u + \frac{\bar{z}^{k-1}}{t} - \frac{\bar{w}^{k-1}}{t} + \frac{1}{N-1}(A_0 x_0^k - b) + \frac{w_i^{k-1}}{t} \right\|^2$$
$$z_i^k = p^k + w_i^{k-1} + t A_i x_i^k$$
$$= \bar{z}^{k-1} - \bar{w}^{k-1} + \frac{t}{N-1}(A_0 x_0^k - b) + t\left(A_i x_i^k + \frac{w_i^{k-1}}{t} \right)$$
$$w_i^k = w_i^{k-1} + p^k - z_i^k = -t A_i x_i^k,$$

for $i = 1, \ldots, N-1$. Notice that the equation for p^k can be omitted. Using the relationship $\frac{w_i^k}{t} = -A_i x_i^k$ we can rewrite this iteration as

$$x_0^k = \underset{u}{\operatorname{argmin}} \quad f_0(u) + \frac{t}{2(N-1)} \left\| A_0 u + \sum_{j=1}^{N-1} A_j x_j^{k-1} - b + \sum_{j=1}^{N-1} \frac{z_j^{k-1}}{t} \right\|^2$$
$$= \underset{u}{\operatorname{argmin}} \quad f_0(u) + \left\langle A_0 u, \bar{z}^{k-1} \right\rangle + \frac{t}{2(N-1)} \left\| A_0 u + \sum_{j=1}^{N-1} A_j x_j^{k-1} - b \right\|^2$$

For $i = 1, \ldots, N-1$:

$$x_i^k = \underset{u}{\operatorname{argmin}}\, f_i(u) + \frac{t}{2} \left\| A_i(u - x_i^{k-1}) + \frac{1}{N-1} \left(A_0 x_0^k + \sum_{j=1}^{N-1} A_j x_j^{k-1} - b \right) + \frac{\bar{z}^{k-1}}{t} \right\|^2$$
$$= \underset{u}{\operatorname{argmin}}\, f_i(u) + \langle A_i u, \bar{z}^{k-1} \rangle + \frac{t}{2} \left\| A_i(u - x_i^{k-1}) + \frac{1}{N-1} \left(A_0 x_0^k + \sum_{j=1}^{N-1} A_j x_j^{k-1} - b \right) \right\|^2$$
$$z_i^k = \bar{z}^{k-1} + \frac{t}{N-1} \left(A_0 x_0^k + \sum_{j=1}^{N-1} A_j x_j^{k-1} - b \right) + t A_i (x_i^k - x_i^{k-1}).$$

This is the extended ADMM iteration. Note that it reduces to the standard ADMM iteration in the case $N = 2$.

5.2.2 Extending ADMM via consensus Douglas-Rachford

Another way to derive an ADMM-like algorithm is to solve (5.9) with the standard consensus Douglas-Rachford. (A similar approach leading to an equivalent iteration is given in [DLP13].) We reformulate (5.9) as

$$\text{minimize} \quad \underbrace{\delta_C(z_0, \ldots, z_{N-1})}_{F(z)} + \underbrace{\sum_{i=0}^{N-1} f_i^*(-A_i^T z_i) + \langle z_i, b_i \rangle}_{G(z)} \tag{5.11}$$

with optimization variable $z = (z_0, \ldots, z_{N-1})$. Here $C = \{(z_0, \ldots, z_{N-1}) \mid z_0 = z_1 = \cdots = z_{N-1}\}$, and δ_C is the indicator function for C. (We have arbitrarily decomposed b as $b = \sum_{i=0}^{N-1} b_i$ in order to make the calculation a little cleaner.) We'll solve (5.11) by Douglas-Rachford.

The Douglas-Rachford iteration for (5.11) is

$$\begin{aligned} p^k &= \text{prox}_{tF}(z^{k-1} - w^{k-1}) \\ z^k &= \text{prox}_{tG}(p^k + w^{k-1}) \\ w^k &= w^{k-1} + p^k - z^k. \end{aligned}$$

The ith block of p^k is $p_i^k = \bar{z}^{k-1} - \bar{w}^{k-1}$, where $\bar{z}^{k-1} = \frac{1}{N}\sum_{i=0}^{N-1} z_i^{k-1}$ and $\bar{w}^{k-1} = \frac{1}{N}\sum_{i=0}^{N-1} w_i^{k-1}$. The ith block of z^k is

$$\begin{aligned} z_i^k &= p_i^k + w_i^{k-1} + t(A_i x_i^k - b_i) \\ &= \bar{z}^{k-1} - \bar{w}^{k-1} + w_i^{k-1} + t(A_i x_i^k - b_i) \\ &= \bar{z}^{k-1} + t\left(-\frac{\bar{w}^{k-1}}{t} + \frac{w_i^{k-1}}{t} + A_i x_i^k - b_i\right), \end{aligned}$$

where

$$x_i^k = \underset{u}{\text{argmin}} \quad f_i(u) + \frac{t}{2}\left\| A_i u - b_i + \frac{\bar{z}^{k-1}}{t} - \frac{\bar{w}^{k-1}}{t} + \frac{w_i^{k-1}}{t} \right\|^2.$$

The ith block of w^k is

$$w_i^k = w_i^{k-1} + p_i^k - z_i^k = -t(A_i x_i^k - b_i).$$

The equation

$$\frac{w_i^k}{t} = -(A_i x_i^k - b_i)$$

can be used to remove w from the iteration. p can also be omitted. The resulting iteration is

$$\begin{aligned}
&\text{For } i = 0, \ldots, N-1: \\
x_i^k &= \underset{u}{\text{argmin}} \quad f_i(u) + \frac{t}{2}\left\| A_i(u - x_i^{k-1}) + \frac{1}{N}\left(\sum_{j=0}^{N-1} A_i x_i^{k-1} - b\right) + \frac{\bar{z}^{k-1}}{t}\right\|^2 \\
&= \underset{u}{\text{argmin}} \quad f_i(u) + \langle A_i u, \bar{z}^{k-1}\rangle + \frac{t}{2}\left\| A_i(u - x_i^{k-1}) + \frac{1}{N}\left(\sum_{j=0}^{N-1} A_i x_i^{k-1} - b\right)\right\|^2 \\
z_i^k &= \bar{z}^{k-1} + t\left(A_i(x_i^k - x_i^{k-1}) + \frac{1}{N}\left(\sum_{j=0}^{N-1} A_i x_i^{k-1} - b\right)\right) \\
&= \bar{z}^{k-1} + \frac{t}{N}\left(\sum_{j=0}^{N-1} A_i x_i^{k-1} - b\right) + tA_i(x_i^k - x_i^{k-1}).
\end{aligned}$$

This is the consensus ADMM iteration. Compared with the extended ADMM iteration derived previously, this approach requires more variables, but has the benefit that all x_i updated are performed in parallel. The consensus ADMM iteration does not reduce to the ADMM iteration in the case $N = 2$. An equivalent iteration is given in [DLP13].

5.3 Experiment

We revisit the L_1 TV deblurring problem of section 3.4.2:

$$\begin{aligned}
&\text{minimize} \quad \|Kx - b\|_1 + \gamma\|Dx\|_{\text{iso}} \\
&\text{subject to} \quad 0 \le x \le 1
\end{aligned} \tag{5.12}$$

where b is a blurry, noisy image (stored as a column vector), K represents a convolution operator using replicate boundary conditions and D represents a discrete gradient operator that uses symmetric boundary conditions. Problem (5.12) has the form

$$\text{minimize} \quad f(x) + g(Ax) \tag{5.13}$$

where f is the indicator function of the set $\{x \mid 0 \leq x \leq 1\}$ and $g(y_1, y_2) = \|y_1 - b\|_1 + \gamma\|y_2\|_{\text{iso}}$, and

$$A = \begin{bmatrix} K \\ D \end{bmatrix}.$$

The matrices K and D can be decomposed as $K = K_p + K_s$ and $D = D_p + D_s$ where K_p and D_p represent convolution operators that use periodic boundary conditions, and K_s and D_s are sparse matrices that correct the values near the boundary. Correspondingly, the matrix A can be decomposed as $A = B + C$, where

$$B = \begin{bmatrix} K_p \\ D_p \end{bmatrix}, \quad C = \begin{bmatrix} K_s \\ D_s \end{bmatrix}.$$

The KKT conditions for problem (5.13), with $A = B + C$, can be expressed as the monotone inclusion problem

$$0 \in \begin{bmatrix} 0 & B^T \\ -B & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} 0 & C^T \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \begin{bmatrix} \partial f(x) \\ \partial g^*(z) \end{bmatrix}. \tag{5.14}$$

We can solve this monotone inclusion problem using the extended Douglas-Rachford method for three operators, taking

$$F_0(x, z) = \begin{bmatrix} \partial f(x) \\ \partial g^*(z) \end{bmatrix}, \quad F_1(x, z) = \begin{bmatrix} 0 & B^T \\ -B & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix}, \quad F_2(x, z) = \begin{bmatrix} 0 & C^T \\ -C & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix}.$$

Another option would be to solve (5.14) by the consensus Douglas-Rachford method. Figure 5.1 compares these methods with the ADMM and primal-dual Douglas-Rachford methods for solving problem (5.12) that were presented in section 3.4.2. Note that the plots for ADMM and primal-dual Douglas-Rachford are identical to those in figure 3.6. Figure 5.1 shows that in this experiment, the consensus Douglas-Rachford method is competitive with the methods of chapter 3, and the extended Douglas-Rachford method compares favorably with the methods of chapter 3.

The average elapsed time per iteration was .024 seconds for Chambolle-Pock, .037 seconds for ADMM, .029 seconds for primal DR, .031 seconds for primal-dual DR, .029 seconds for

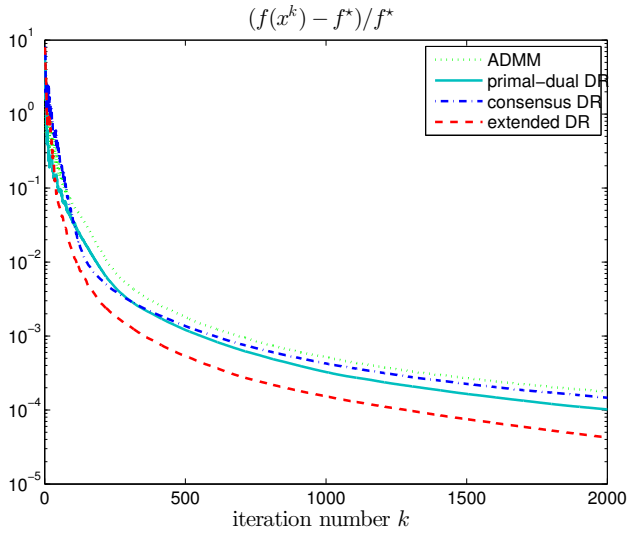


Figure 5.1: Relative optimality gap versus iteration number for the experiment in section 5.3. Compare this figure with figure 3.4.2.

consensus DR, and .028 seconds for extended DR. Close to optimal step sizes for all methods were selected by trial and error. This experiment was performed on a computer with a 3.70 GHz Intel Xeon(R) E5-1620 v2 processor with 8 cores and 7.7 GB of RAM. The code was written in MATLAB using MATLAB version 8.1.0.604 (R2013a).

5.4 A domain decomposition approach to image deblurring

In this section we'll derive a domain decomposition image deblurring algorithm that makes use of the extended Douglas-Rachford algorithm described above.

5.4.1 "Free" boundary conditions for image deblurring

We first explain in more detail the "unknown boundary conditions" approach to handling non-periodic boundary conditions in image deblurring [AF13], which was discussed briefly in section 4.2.

Suppose we want to deblur an image $b \in \mathbb{R}^{M \times N}$, and the (space-invariant) blur kernel

κ is known and has support of size $(2d+1) \times (2d+1)$. We first enlarge b to an image $\hat{b} \in \mathbb{R}^{(M+2d)\times(N+2d)}$ by adding to b a border of width d pixels, filled with all zeros. We then solve

$$\begin{aligned} \underset{x}{\text{minimize}} \quad & \frac{1}{2}\|EKx - \hat{b}\|^2 + \gamma\|Dx\|_1 \\ \text{subject to} \quad & 0 \leq x \leq 1 \end{aligned} \tag{5.15}$$

where:

- The optimization variable $x \in \mathbb{R}^{(M+2d)\times(N+2d)}$. Note that in this section it seems more clear to let x be a matrix, whereas previously in this thesis x has been a vector.
- K is a linear operator on $\mathbb{R}^{(M+2d)\times(N+2d)}$ that performs a convolution using the kernel κ and *periodic* boundary conditions .
- E is a linear operator on $\mathbb{R}^{(M+2d)\times(N+2d)}$ that simply zeros out "border" components and leaves all other components unchanged .
- D is a tight frame analysis operator.

The final deblurred image is obtained by removing the border from x. (We could equally well handle other common deblurring optimization formulations; we focus on (5.15) just for clarity.)

Problem (5.15) can be solved efficiently by Douglas-Rachford-based methods because the prox-operator of the function

$$g_1(y_1) = \frac{1}{2}\|Ey_1 - \hat{b}\|^2$$

is inexpensive to evaluate.

Problem (5.15) can be interpreted as a combined deblurring and inpainting problem. The operator E could also zero out pixels corresponding to missing measurements, for example if text has been printed on top of the image b (as in many inpainting examples). In this case, the corresponding pixels of $\hat{b}$, where we have missing or invalid data, should be zeroed out as well.

5.4.2 Domain decomposition approach

Now we'll describe a domain decomposition approach to image deblurring, inspired by the paper [CZT13].

We view the blurry, noisy image b as being composed of $\mathcal{I}\mathcal{J}$ subimages b_{ij}, each of size $m \times n$. For each subimage b_{ij}, the zero-padded image $\hat{b}$ (see section 5.4.1) has a corresponding subimage $\hat{b}_{ij}$ of size $(m+2d) \times (n+2d)$. (Note that the subimages of $\hat{b}$ overlap. See figure 3.1 in [CZT13].) Each subimage b_{ij} could be obtained from $\hat{b}_{ij}$ just by removing the "border" of width d pixels.

Here is some useful notation:

- Let k be the linear operator on $\mathbb{R}^{(m+2d)\times(n+2d)}$ that performs a convolution using the kernel κ and periodic boundary conditions.
- Let K_{ij} be the linear operator on $\mathbb{R}^{(M+2d)\times(N+2d)}$ that, when given an input x, applies k to the subimage $x_{ij} \in \mathbb{R}^{(m+2d)\times(n+2d)}$ and zeros out all other components of x.
- Let E_{ij} be the linear operator on $\mathbb{R}^{(M+2d)\times(N+2d)}$ that, given an input x, leaves the non-border pixels of the subimage $x_{ij} \in \mathbb{R}^{(m+2d)\times(n+2d)}$ untouched, but zeros out the border pixels of x_{ij}, and also zeros out all other components of x.

Note that

$$EK = \sum_{i,j} E_{ij} K_{ij}.$$

Here's some additional notation: Let

$$\begin{aligned}
S &= \{(i,j) \mid 1 \le i \le \mathcal{I}, 1 \le j \le \mathcal{J}\}, \\
S_1 &= \{(i,j) \in S \mid i \text{ and } j \text{ are odd}\}, \\
S_2 &= \{(i,j) \in S \mid i \text{ is odd and } j \text{ is even}\}, \\
S_3 &= \{(i,j) \in S \mid i \text{ is even and } j \text{ is odd}\}, \text{ and} \\
S_4 &= \{(i,j) \in S \mid i \text{ and } j \text{ are even}\}.
\end{aligned}$$

Furthermore, let

$$K_\ell = \sum_{(i,j)\in S_\ell} K_{ij}, \qquad E_\ell = \sum_{(i,j)\in S_\ell} E_{ij}$$

for $\ell = 1, 2, 3, 4$. A key fact is that

$$EK = E_1K_1 + E_2K_2 + E_3K_3 + E_4K_4.$$

Problem (5.15) can be expressed in the standard form

$$\underset{x}{\text{minimize}} \quad f(x) + g(Ax) \tag{5.16}$$

where f is the indicator function of $\{x \mid 0 \le x \le 1\}$,

$$g(u_1, u_2, u_3, u_4, w) = g_1(u_1, u_2, u_3, u_4) + g_2(w)$$

with

$$g_1(u_1, u_2, u_3, u_4) = \frac{1}{2}\|E_1u_1 + E_2u_2 + E_3u_3 + E_4u_4\|^2$$

and $g_2(w) = \gamma\|w\|_1$, and

$$A = \begin{bmatrix} K_1 \\ K_2 \\ K_3 \\ K_4 \\ D \end{bmatrix}.$$

Note that

$$A = A_1 + A_2 + A_3 + A_4$$

where

$$A_1 = \begin{bmatrix} K_1 \\ 0 \\ 0 \\ 0 \\ D \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 \\ K_2 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 0 \\ 0 \\ K_3 \\ 0 \\ 0 \end{bmatrix}, \quad A_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ K_4 \\ 0 \end{bmatrix}.$$

(a) Blurry, noisy image.

(b) After 65 iterations.

(c) Restored image.

Each of the transformations A_p (for $p = 1, \ldots, 4$) is "easy" in the sense that equations with coefficient $I + \lambda A_p^T A_p$ can be solved efficiently via the FFT.

The optimality conditions (2.11) for optimization problem (5.16) can be written as

$$0 \in \begin{bmatrix} \partial f(x) \\ \partial g^*(z) \end{bmatrix} + \sum_{i=1}^{4} \begin{bmatrix} 0 & A_i^T \\ -A_i & 0 \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix}. \tag{5.17}$$

The monotone operator on the right is a sum of five "easy" operators, therefore we can solve the monotone inclusion problem (5.17) using either the consensus Douglas-Rachford method or the extended Douglas-Rachford method from section 5.1.

5.4.3 Experiment

Figure 5.4.3 shows a deblurring result obtained using the domain decomposition approach described in the previous section. The image b is a degraded version of the 512 by 512 "Barbara" image. The original image (scaled so that intensity values are between 0 and 1) was blurred with a 17 by 17 truncated Gaussian kernel with standard deviation $\sigma = 3$. Gaussian noise with zero mean and standard deviation 10^{-3} was added to each pixel of the blurred image, then salt and pepper noise was added to a random selection of 10% of the pixels. We took $\mathcal{I} = \mathcal{J} = 4$, so that b is viewed as being composed of 16 subimages. The parameter γ was chosen to give a visually appealing image reconstruction. The Douglas-Rachford step size and overrelaxation parameter were selected by trial and error. In this

experiment, total variation regularization was used rather than tight frame regularization, and A was decomposed as a sum of five terms rather than four: $A = A_1 + A_2 + A_3 + A_4 + A_5$ where

$$A_1 = \begin{bmatrix} K_1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0 \\ K_2 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad A_3 = \begin{bmatrix} 0 \\ 0 \\ K_3 \\ 0 \\ 0 \end{bmatrix}, \quad A_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ K_4 \\ 0 \end{bmatrix}, \quad A_5 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ D \end{bmatrix}$$

and D is a discrete gradient operator.

CHAPTER 6

Conclusions

We have presented Douglas-Rachford-based methods for solving convex optimization problems of the canonical form

$$\underset{x}{\text{minimize}} \quad f(x) + g(Ax), \tag{6.1}$$

where f and g are closed convex functions that are "simple" in the sense that their proximal operators can be evaluated efficiently. In the most basic case, the matrix A has structure that allows linear equations with coefficient matrix $I + \lambda A^T A$ to be solved efficiently (where λ is a positive constant). In chapter 3 we presented algorithms for the case where A is a sum of two structured matrices: $A = B + C$, where linear systems with coefficient matrix $I + \lambda B^T B$ or $I + \lambda C^T C$ (but not $I + \lambda A^T A$) can be solved efficiently. Chapter 5 described a primal-dual operator splitting approach to solving the canonical problem in the case where A is a sum of N structured matrices. The methods of chapter 5 are based on using consensus Douglas-Rachford or an extended Douglas-Rachford method to solve the primal-dual optimality conditions, which are expressed as a monotone inclusion problem involving a sum of $N + 1$ monotone operators.

In chapter 3 we gave applications to image deblurring, discussing how the general deblurring problem

$$\text{minimize} \quad \phi_{\mathrm{f}}(Kx - b) + \phi_{\mathrm{r}}(x) + \phi_{\mathrm{s}}(Dx)$$

can be expressed in the canonical form (6.1) and how physically realistic boundary conditions (such as replicate boundary conditions) for the blur operator K can be handled by using techniques that exploit additive structure in A. Although these optimization problems are very large scale, constrained, and have non-differentiable objective functions, the Douglas-

Rachford-based methods presented in this thesis yield simple iterative algorithms where at each iteration we only have to compute a small number of fast Fourier transforms, evaluate proximal operators with complexity linear in the number of variables, and solve a very sparse system of linear equations. Despite the necessity of solving linear systems at each iteration, the Douglas-Rachford-based methods are able (by exploiting structure in K and D) to achieve a per-iteration complexity similar to that of the Chambolle-Pock algorithm (which requires multiplications by A or A^T at each iteration, but does not require solving linear systems involving A). In chapter 5, we presented a domain decomposition approach to image deblurring, where the image is partitioned into rectangular patches and the blur operator is written as a sum of four terms, each term corresponding to a set of non-overlapping patches. In this case the primal-dual optimality conditions are expressed as a monotone inclusion problem involving a sum of five simple monotone operators, and this inclusion problem is solved using the consensus or extended Douglas-Rachford methods. The resulting algorithm has the advantage that at each iteration, all patches are processed in parallel, and the blur operator is able to use a different blur kernel for each patch. In chapter 4, we considered the problem of total variation or tight frame regularized image deblurring using two fundamental models of spatially variant blur: the Nagy-O'Leary model and the related Efficient Filter Flow model. We showed how, in each case, the deblurring problem can be expressed in the canonical form with simple choices of f and g and structured A, so that Douglas-Rachford based methods can be applied. In each case, the key steps are the definitions of g and A and the computation of the prox-operator of g. The Douglas-Rachford-based methods are able to solving spatially variant deblurring problems with a complexity comparable to that of algorithms for spatially invariant deblurring. The cost of each iteration is dominated by the computation of a small number of fast Fourier transforms.

The basic approach of the primal-dual operator splitting methods presented in this thesis has been to express the primal-dual optimality conditions (KKT conditions) as a monotone inclusion problem, which is then solved using the Douglas-Rachford method. This approach is motivated by the fact that f, g, and A appear separately in the primal-dual optimality

conditions, which makes it straightforward to exploit additive structure in A. The purely primal and dual approaches, on the other hand, may require a large number of splitting variables and extra constraints to exploit the same type of structure.

In this thesis we have seen that Douglas-Rachford-based methods can often be implemented to have the same complexity per iteration as semi-implicit methods such as the Chambolle-Pock method. An advantage of the Douglas-Rachford-based methods is that they do not have a limit on the step size, and do not require estimating the norm of A.

Image deblurring problems are challenging because they are large scale, constrained, and nondifferentiable, and for that reason they serve as good test problems for the algorithms presented in this thesis. One direction of future work is to apply these techniques to image reconstruction problems and related problems arising in areas such as medical imaging and medical physics. For example, it may be possible to derive spatially variant blur models for OCT or ultrasound imaging and to apply the deblurring techniques of chapter 4. Beyond image processing, many large scale problems in areas such as machine learning, statistics, signal processing, and finance are conveniently expressed in the canonical form (6.1) and applying the decomposition algorithms of this thesis to problems arising in these application areas could be a subject of much future work.

References

[AF13] M. S. C. Almeida and M. A. T. Figueiredo. "Deconvolving images with unknown boundaries using the alternating direction method of multipliers." *IEEE Trans. Image Process.*, **22**(8):3074–3086, 2013.

[BB11] S. Ben Hadj and L. Blanc Féraud. "Restoration method for spatially variant blurred images." Rapport de recherche RR-7654, INRIA, June 2011.

[BB12] S. Ben Hadj and L. Blanc-Féraud. "Modeling and removing depth variant blur in 3D fluorescence microscopy." In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pp. 689–692. IEEE, 2012.

[BBA12] S. Ben Hadj, L. Blanc-Féraud, G. Aubert, et al. "Space variant blind image restoration." 2012.

[BC11a] H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2011.

[BC11b] L. M. Briceño-Arias and P. L. Combettes. "A monotone + skew splitting model for composite monotone inclusions in duality." *SIAM Journal on Optimization*, **21**(4):1230–1250, 2011.

[BJN06] J. Bardsley, S. Jefferies, J. Nagy, and R. Plemmons. "A computational method for the restoration of images with an unknown, spatially-varying blur." *Optics express*, **14**(5):1767–1782, 2006.

[BN01] A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization. Analysis, Algorithms, and Engineering Applications*. Society for Industrial and Applied Mathematics, 2001.

[BPC11] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. "Distributed optimization and statistical learning via the alternating direction method of multipliers." *Foundations and Trends in Machine Learning*, **3**(1):1–122, 2011.

[Bre73] H. Brézis. *Opérateurs maximaux monotones et semi-groupes de contractions dans les espaces de Hilbert*, volume 5 of *North-Holland Mathematical Studies*. North-Holland, 1973.

[BT89] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation*. Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

[BT09] A. Beck and M. Teboulle. "Fast gradient-based algorithm for contrained total variation image denoising and deblurring problems." *IEEE Transactions on Image Processing*, **18**(11):2419–2434, 2009.

[CDS01] S. S. Chen, D. L. Donoho, and M. A. Saunders. "Atomic decomposition by basis pursuit." *SIAM Review*, **43**(1):129–159, 2001.

[CGM99] T. F. Chan, G. H. Golub, and P. Mulet. "A nonlinear primal-dual method for total variation-based image restoration." *SIAM Journal on Scientific Computing*, **20**(6):1964–1977, 1999.

[CL09] S. Cho and S. Lee. "Fast Motion Deblurring." *ACM Trans. Graph.*, **28**(5):145:1–145:8, December 2009.

[CLC07] M. Chiang, S. H. Low, R. Calderbank, and J. C. Doyle. "Layering as optimization decomposition: a mathematical theory of network architectures." *Proceedings of the IEEE*, **95**(1):255–312, 2007.

[Con13] L. Condat. "A primal-dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms." *Journal of Optimization Theory and Applications*, **158**(2):460–479, 2013.

[CP07] P. L. Combettes and J.-C. Pesquet. "A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery." *IEEE Journal of Selected Topics in Signal Processing*, **1**(4):564–574, 2007.

[CP10] P. L. Combettes and J.-Ch. Pesquet. "Proximal splitting methods in signal processing." 2010. Available from `arxiv.org/abs/0912.3522v4`.

[CP11a] A. Chambolle and T. Pock. "A first-order primal-dual algorithms for convex problems with applications to imaging." *Journal of Mathematical Imaging and Vision*, **40**:120–145, 2011.

[CP11b] P. L. Combettes and J.-C. Pesquet. "Proximal Splitting Methods in Signal Processing." In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, Springer Optimization and Its Applications, pp. 185–212. Springer New York, 2011.

[CP12] P. L. Combettes and J.-C. Pesquet. "Primal-dual splitting algorithm for solving inclusions with mixtures of composite, Lipschitzian, and parallel-sum type monotone operators." *Set-Valued and Variational Analysis*, **20**(2):307–330, 2012.

[CTY13] R. H. Chan, M. Tao, and X. Yuan. "Constrained total variation deblurring methods and fast algorithms based on alternating direction method of multipliers." *SIAM Journal on Imaging Sciences*, **6**(1):680–697, 2013.

[CW05] P. L. Combettes and V. R. Wajs. "Signal recovery by proximal forward-backward splitting." *Multiscale Modeling and Simulation*, **4**(4):1168–1200, 2005.

[CYP05] T.F. Chan, A.M. Yip, and F.E. Park. "Simultaneous total variation image inpainting and blind deconvolution." *International Journal of Imaging Systems and Technology*, **15**(1):92–102, 2005.

[CZF10] A. Chakrabarti, T. Zickler, and W. T. Freeman. "Analyzing spatially-varying blur." In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 2512–2519. IEEE, 2010.

[CZT13] Huibin Chang, Xiaoqun Zhang, Xue-Cheng Tai, and Danping Yang. "Domain Decomposition Methods for Nonlocal Total Variation Image Restoration." *Journal of Scientific Computing*, pp. 1–22, 2013.

[DLP13] W. Deng, M. Lai, Z. Peng, and W. Yin. "Parallel multi-block ADMM with $O(1/k)$ convergence." *arXiv preprint arXiv:1312.3040*, 2013.

[DTS11] L. Denis, E. Thiebaut, and F. Soulez. "Fast model of space-variant blurring and its application to deconvolution in astronomy." In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pp. 2817–2820, Sept 2011.

[DY14] D. Davis and W. Yin. "Convergence rate analysis of several splitting schemes." *arXiv preprint arXiv:1406.4834*, 2014.

[EB92] J. Eckstein and D. Bertsekas. "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators." *Mathematical Programming*, **55**:293–318, 1992.

[Eck12] J. Eckstein. "Augmented Lagrangian and alternating minimization methods for convex optimization: a tutorial and some illustrative computational results." Technical Report RRR 32-2012, Rutgers Center for Operations Research, 2012.

[Ess10] J. E. Esser. *Primal Dual Algorithms for Convex Models and Applications to Image Restoration, Registration and Nonlocal Inpainting.* PhD thesis, Department of Mathematics, UCLA, 2010.

[EZC09] E. Esser, X. Zhang, and T. Chan. "A general framework for a class of first order primal-dual algorithms for TV minimization." Technical Report CAM 09-67, UCLA Department of Mathematics, 2009.

[FLS10] M. Fornasier, A. Langer, and C.-B. Schnlieb. "A convergent overlapping domain decomposition method for total variation minimization." *Numerische Mathematik*, **116**(4):645–685, 2010.

[Gab83] D. Gabay. "Applications of the method of multipliers to variational inequalities." In M. Fortin and R. Glowinski, editors, *Augmented Lagrangian methods: Applications to the numerical solution of boundary-value problems*, Studies in Mathematics and Its Applications, pp. 299–331. North-Holland, 1983.

[GB08] M. Grant and S. Boyd. "Graph implementations for nonsmooth convex programs." In V. Blondel, S. Boyd, and H. Kimura, editors, *Recent Advances in Learning and Control (a tribute to M. Vidyasagar)*, pp. 95–110. Springer, 2008.

[GB12] M. Grant and S. Boyd. *CVX: Matlab Software for Disciplined Convex Programming, version 2.0 (beta).* `cvxr.com`, 2012.

[GM75] R. Glowinski and A. Marrocco. "Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de Dirichlet non linéaires." *Revue française d'automatique, informatique, recherche opérationnelle*, **9**(2):41–76, 1975.

[GM76] D. Gabay and B. Mercier. "A dual algorithm for the solution of nonlinear variational problems via finite element approximation." *Computers and Mathematics with Applications*, **2**:17–40, 1976.

[GO09] T. Goldstein and S. Osher. "The split Bregman method for L1-regularized problems." *SIAM Journal on Imaging Sciences*, **2**(2):323–343, 2009.

[HHS10] S. Harmeling, M. Hirsch, and B. Schölkopf. "Space-variant single-image blind deconvolution for removing camera shake." In J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pp. 829–837. Curran Associates, Inc., 2010.

[HLW03] B. S. He, L. Z. Liao, and S. L. Wang. "Self-adaptive operator splitting methods for monotone variational inequalities." *Numerische Mathematik*, **94**:715–737, 2003.

[HNO06] P. C. Hansen, J. G. Nagy, and D. P. O'Leary. *Deblurring Images. Matrices, Spectra, and Filtering.* Society for Industrial and Applied Mathematics, 2006.

[HSH11] M. Hirsch, C. J. Schuler, S. Harmeling, and B. Scholkopf. "Fast Removal of Non-uniform Camera Shake." In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pp. 463–470, Washington, DC, USA, 2011. IEEE Computer Society.

[HSS10] M. Hirsch, S. Sra, B. Scholkopf, and S. Harmeling. "Efficient filter flow for space-variant multiframe blind deconvolution." In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 607–614, June 2010.

[HY12] B. He and X. Yuan. "Convergence analysis of primal-dual algorithms for a saddle-point problem: from contraction perspective." *SIAM Journal on Imaging Sciences*, **5**(1):119–149, 2012.

[HYW00] B. S. He, H. Yang, and S. L. Wang. "Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities." *Journal of Optimization Theory and Applications*, **106**:337–356, 2000.

[JKZ10] N. Joshi, S. B. Kang, C. L. Zitnick, and R. Szeliski. "Image Deblurring Using Inertial Measurement Sensors." *ACM Trans. Graph.*, **29**(4):30:1–30:9, July 2010.

[Kar84] N. Karmarkar. "A new polynomial-time algorithm for linear programming." In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pp. 302–311. ACM, 1984.

[KL12] G. Kutyniok and D. Labate. *Shearlets: Multiscale analysis for multivariate data*. Springer, 2012.

[KSZ11] G. Kutyniok, M. Shahram, and X. Zhuang. "Shearlab: A rational design of a digital parabolic scaling algorithm." *arXiv preprint arXiv:1106.1319*, 2011.

[Las70] L. S. Lasdon. *Optimization Theory for Large Systems*. MacMillan, 1970.

[Lev06] A. Levin. "Blind motion deblurring using image statistics." In *NIPS*, volume 2, p. 4, 2006.

[LM79] P. L. Lions and B. Mercier. "Splitting algorithms for the sum of two nonlinear operators." *SIAM Journal on Numerical Analysis*, **16**(6):964–979, 1979.

[Mal99] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, second edition, 1999.

[Mor65] J. J. Moreau. "Proximité et dualité dans un espace hilbertien." *Bull. Math. Soc. France*, **93**:273–299, 1965.

[NCT99] M. K. Ng, R. H. Chan, and W.-C. Tang. "A fast algorithm for deblurring models with Neumann boundary conditions." *SIAM Journal on Scientific Computing*, **21**(3):851–866, 1999.

[NN94] Yu. Nesterov and A. Nemirovskii. *Interior-point polynomial methods in convex programming*, volume 13 of *Studies in Applied Mathematics*. SIAM, Philadelphia, PA, 1994.

[NO98] J. G. Nagy and D. P. O'Leary. "Restoring images degraded by spatially variant blur." *SIAM Journal on Scientific Computing*, **19**(4):1063–1082, 1998.

[OV] D. O'Connor and L. Vandenberghe. "Total variation image deblurring with space-varying kernel.".

[OV14] D. O'Connor and L. Vandenberghe. "Primal-dual decomposition by operator splitting and applications to image deblurring." *SIAM Journal on Imaging Sciences*, **7**:1724–1754, 2014.

[PB13] N. Parikh and S. Boyd. "Proximal algorithms." *Foundations and Trends in Optimization*, **1**(3):123–231, 2013.

[PC04] C. Preza and J.-A. Conchello. "Depth-variant maximum-likelihood restoration for three-dimensional fluorescence microscopy." *JOSA A*, **21**(9):1593–1601, 2004.

[PC06] D. P. Palomar and M. Chiang. "A tutorial on decomposition methods for network utility maximization." *IEEE Journal on Selected Areas in Communications*, **24**:1439–1451, 2006.

[RO94] L.I. Rudin and S. Osher. "Total variation based image restoration with free local constraints." In *Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference*, volume 1, pp. 31–35 vol.1, Nov 1994.

[Roc67] R.T. Rockafellar. "Duality and stability in extremum problems involving convex functions." *Pacific Journal of Mathematics*, **21**(1):167–187, 1967.

[Roc70] R. T. Rockafellar. *Convex Analysis*. Princeton Univ. Press, Princeton, second edition, 1970.

[Roc74] R. T. Rockafellar. *Conjugate Duality and Optimization*, volume 16 of *Regional Conference Series in Applied Mathematics*. SIAM, 1974.

[Roc76a] R. T. Rockafellar. "Augmented Lagrangians and applications of the proximal point algorithm in convex programming." *Mathematics of Operations Research*, **1**(2):97–116, 1976.

[Roc76b] R. T. Rockafellar. "Monotone Operators and the Proximal Point Algorithm." *SIAM J. Control and Opt.*, **14**(5):877–898, August 1976.

[ROF92] L. Rudin, S. J. Osher, and E. Fatemi. "Nonlinear total variation based noise removal algorithms." *Physica D*, **60**:259–268, 1992.

[Sor12] M. Šorel. "Removing boundary artifacts for real-time iterated shrinkage deconvolution." *IEEE Transactions on Image Processing*, **21**(4):2329, 2012.

[Spi83] J. E. Spingarn. "Partial inverse of a monotone operator." *Applied Mathematics and Optimization*, **10**:247–265, 1983.

[Spi85] J. E. Spingarn. "Applications of the method of partial inverses to convex programming: decomposition." *Mathematical Programming*, **32**:199–223, 1985.

[Str87] G. Strang. "Karmarkars algorithm and its place in applied mathematics." *The Mathematical Intelligencer*, **9**(2):4–10, 1987.

[Tse00] P. Tseng. "A modified forward-backward splitting method for maximal monotone mappings." *SIAM Journal on Control and Optimization*, **38**(2):431–446, 2000.

[Vog02] C. R. Vogel. *Computational Methods for Inverse Problems*. Society for Industrial and Applied Mathematics, 2002.

[Wri97] S. J. Wright. *Primal-Dual Interior-Point Methods*. SIAM, Philadelphia, 1997.

[WSZ12] O. Whyte, J. Sivic, A. Zisserman, and J. Ponce. "Non-uniform deblurring for shaken images." *International Journal of Computer Vision*, **98**(2):168–186, 2012.

[YOG08] W. Yin, S. Osher, D. Goldfarb, and J. Darbon. "Bregman iterative algorithms for ℓ_1-minimization with application to compressed sensing." *SIAM J. Imaging Sciences*, **1**(1), 2008.