Diagonal preconditioning for first order primal-dual algorithms in convex optimization*

Thomas Pock
Institute for Computer Graphics and Vision
Graz University of Technology
pock@icg.tugraz.at

Antonin Chambolle
CMAP & CNRS
École Polytechnique
antonin.chambolle@cmap.polytechnique.fr

Abstract

In this paper we study preconditioning techniques for the first-order primal-dual algorithm proposed in [5]. In particular, we propose simple and easy to compute diagonal preconditioners for which convergence of the algorithm is guaranteed without the need to compute any step size parameters. As a by-product, we show that for a certain instance of the preconditioning, the proposed algorithm is equivalent to the old and widely unknown alternating step method for monotropic programming [7]. We show numerical results on general linear programming problems and a few standard computer vision problems. In all examples, the preconditioned algorithm significantly outperforms the algorithm of [5].

1. Introduction

In [5, 8, 13] first-order primal-dual algorithms are studied to solve a certain class of convex optimization problems with known saddle-point structure.

$$\min_{x\in X}\max_{y\in Y}\ \langle Kx,y\rangle + G(x) - F^*(y)\ , \tag{1}$$

where X and Y are finite-dimensional vector spaces equipped with standard inner products $\langle\cdot,\cdot\rangle$. $K : X \to Y$ is a linear operator and $G : X \to \mathbb{R}\cup\{\infty\}$ and $F^* : Y \to \mathbb{R}\cup\{\infty\}$ are convex functions with known structure.

The iterates of the algorithm studied in [5] to solve (1) are very simple:

$$\begin{cases} x^{k+1} = (I+\tau\partial G)^{-1}(x^k - \tau K^T y^k) \\ y^{k+1} = (I+\sigma\partial F^*)^{-1}(y^k + \sigma K(x^{k+1} + \theta(x^{k+1} - x^k))) \end{cases} \tag{2}$$

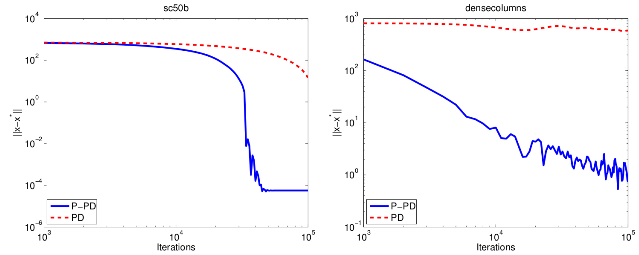


Figure 1. On problems with irregular structure, the proposed preconditioned algorithm (P-PD) converges significantly faster than the algorithm of [5] (PD).

They basically consist of alternating a gradient ascend in the dual variable and a gradient descend in the primal variable. Additionally, the algorithm performs an overrelaxation step in the primal variable. A fundamental assumption of the algorithm is that the functions F^* and G are of simple structure, meaning that the so-called proximity or resolvent operators $(I+\sigma\partial F^*)^{-1}$ and $(I+\tau\partial G)^{-1}$ have closed-form solutions or can be efficiently computed with a high precision. Their exact definitions will be given in Section 1.1. The parameters $\tau,\sigma > 0$ are the primal and dual step sizes and $\theta\in[0,1]$ controls the amount of overrelaxation in x. It is shown in [5] that the algorithm converges as long as $\theta = 1$ and the primal and dual step sizes τ and σ are chosen such that $\tau\sigma L^2 < 1$, where $L = \|K\|$ is the operator norm of K. It is further shown that a suitably defined partial primal-dual gap of the average of the sequence $((x^0,y^0),...,(x^k,y^k))$ vanishes with rate $O(1/k)$ for the complete class of problems covered by (1). For problems with more regularity, the authors propose acceleration schemes based on non-empirical choices on τ, σ and θ. In particular they show that they can achieve $O(1/k^2)$ for problems where G of F^* is uniformly convex and $O(\omega^k)$, $\omega < 1$ for problems where both G and F^* are uniformly convex. See [5] for more details.

A common feature of all numerical examples in [5] is that the involved linear operators K have a simple structure which makes it very easy to estimate L. We observed that for problems where the operator K has a more compli-

*The first author acknowledges support from the Austrian Science Fund (FWF) under the grant P22492-N23.

cated structure, L cannot be estimated easily or it might be very large such that the convergence of the algorithm significantly slows down. As we will see, linear operators with irregular structure frequently arise in many different vision problems.

In this work, we study preconditioning techniques for the primal-dual algorithm (2). This allows us to overcome the aforementioned shortcomings. The proposed preconditioned algorithm has several advantages. Firstly, it avoids the estimation of the operator norm of K, secondly, it significantly accelerates the convergence on problems with irregular K and thirdly, it leaves the computational complexity of the iterations basically unchanged. Figure 1 shows convergence plots on two LP problems with such an irregular structure. The proposed algorithm can better adapt to the problem structure, leading to faster convergence.

The rest of the paper is as follows. In Section 1.1 we fix some preliminary definitions which will be used throughout the paper. In Section 2 we present the preconditioned primal-dual algorithm and give conditions under which convergence of the algorithm is guaranteed. We propose a family of simple and easy to compute diagonal preconditioners, which turn out to be very efficient on many problems. In Section 2.3 we establish connections to the old and widely unknown alternating step method for monotropic programming [7]. In Section 3 we detail experimental results of the proposed algorithm. In the last Section we draw some conclusions and show directions for future work.

1.1. Preliminaries

We consider finite-dimensional vector spaces X and Y, where $n = \dim X$ and $m = \dim Y$ with inner products

$$\begin{aligned} \left\langle x^1, x^2 \right\rangle_X &= \left\langle \mathrm{T}^{-1} x^1, x^2 \right\rangle , \quad x_1, x_2 \in X , \\ \left\langle y^1, y^2 \right\rangle_Y &= \left\langle \Sigma^{-1} y^1, y^2 \right\rangle , \quad y_1, y_2 \in Y , \end{aligned}$$

where T and Σ are a symmetric, positive definite preconditioning matrices. We further define the norms in the usual way as

$$\|x\|_X = \langle x, x \rangle_X^{\frac{1}{2}} \ , \quad \|y\|_Y = \langle y, y \rangle_Y^{\frac{1}{2}} \ .$$

We will make frequent use of the so-called resolvent or proximity operator of a function $G(x)$. Given a point $\hat{x} \in X$, it is defined as the solution of the auxiliary minimization problem

$$x^* = \arg\min_x G(x) + \frac{1}{2}\|x - \hat{x}\|_X^2$$

The unique minimizer to the above problem is characterized by the optimality condition

$$\partial G(x) + \mathrm{T}^{-1}(x - \hat{x}) \ni 0 \ ,$$

whose optimal solution x^* can be written in operator form as

$$x^* = (I + \mathrm{T}\partial G)^{-1}(\hat{x}) \ . \tag{3}$$

2. Preconditioned primal-dual algorithm

In this work, we propose the following preconditioned first-order primal-dual algorithm: Choose symmetric and positive definite matrices T, Σ, $\theta \in [0, 1]$, $(x^0, y^0) \in X \times Y$. Then for $k \geq 0$, update (x^k, y^k) as follows:

$$\begin{cases} x^{k+1} = (I + \mathrm{T}\partial G)^{-1}(x^k - \mathrm{T}K^T y^k) \\ y^{k+1} = (I + \Sigma\partial F^*)^{-1}(y^k + \Sigma K(x^{k+1} + \theta(x^{k+1} - x^k))) \end{cases} \tag{4}$$

Comparing the iterates (4) of the proposed algorithm to (2), one can see that the global steps τ and σ have been replaced by the preconditioning matrices T and Σ. It is known that (2) converges as long as $\theta = 1$ and $\tau\sigma\|K\|^2 < 1$. Hence, a natural question is now to establish conditions on T and Σ and θ which ensure convergence of the proposed preconditioned algorithm. In very recent work [10], it has been shown that the iterates (2) can be written in form of a proximal point algorithm [14], which greatly simplifies the convergence analysis.

From the optimality conditions of the iterates (4) and the convexity of G and F^* it follows that for any $(x, y) \in X \times Y$ the iterates x^{k+1} and y^{k+1} satisfy

$$\left\langle \begin{pmatrix} x - x^{k+1} \\ y - y^{k+1} \end{pmatrix}, F\begin{pmatrix} x^{k+1} \\ y^{k+1} \end{pmatrix} + M\begin{pmatrix} x^{k+1} - x^k \\ y^{k+1} - y^k \end{pmatrix} \right\rangle \geq 0 \ , \tag{5}$$

where

$$F\begin{pmatrix} x^{k+1} \\ y^{k+1} \end{pmatrix} = \begin{pmatrix} \partial G(x^{k+1}) + K^T y^{k+1} \\ \partial F^*(y^{k+1}) - K x^{k+1} \end{pmatrix} \ ,$$

and

$$M = \begin{bmatrix} \mathrm{T}^{-1} & -K^T \\ -\theta K & \Sigma^{-1} \end{bmatrix} \ . \tag{6}$$

It is easy to check, that the variational inequality (5) now takes the form of a proximal point algorithm [10, 14, 16]. In the next Section we will establish conditions on θ, T and Σ which ensure convergence of the algorithm.

2.1. Convergence of the algorithm

We can make direct use of the convergence analysis developed in [10, 14, 16]. In fact, convergence of (5) can be guaranteed as long as the matrix M is symmetric and positive definite. In the following Lemma we establish conditions on θ, T and Σ which indeed ensure these properties of M.

Lemma 1. *Let $\theta = 1$, T and Σ symmetric positive definite maps satisfying*

$$\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 < 1 \ , \tag{7}$$

then the matrix M defined in (6) *is symmetric and positive definite, more precisely it holds*

$$\left\langle (x,y)^T, M(x,y)^T \right\rangle > 0 \;, \tag{8}$$

for any $(x,y)^T \in X \times Y$ and $(x,y)^T \neq 0$.

Proof. Due to the structure of M, symmetry follows directly from the symmetry of T, Σ and the fact that $\theta = 1$.

Expanding the inner product in (8) we have

$$\begin{aligned} \left\langle \mathrm{T}^{-1}x, x \right\rangle + \left\langle \Sigma^{-1}y, y \right\rangle - 2\left\langle Kx, y \right\rangle &> 0 \\ \|x\|_X^2 + \|y\|_Y^2 - 2\left\langle Kx, y \right\rangle &> 0 \;. \end{aligned} \tag{9}$$

Since T and Σ are symmetric and positive definite, the above scalar product can be rewritten as

$$-2\left\langle Kx, y \right\rangle \;=\; -2\left\langle \Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} \mathrm{T}^{-\frac{1}{2}} x, \Sigma^{-\frac{1}{2}} y \right\rangle$$

Using the Cauchy-Schwarz inequality and the fact that $2ab \leq ca^2 + b^2/c$ for any a, b and $c > 0$, we obtain

$$\begin{aligned} -2\left\langle Kx, y \right\rangle &\geq -2\|\Sigma^{-\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} \mathrm{T}^{-\frac{1}{2}} x\| \|\Sigma^{-\frac{1}{2}} y\| \\ &\geq -\left(c\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 \|x\|_X^2 + \frac{1}{c}\|y\|_Y^2 \right) \end{aligned}$$

In view of (7), it is clear that there exists a $\varepsilon > 0$ such that it holds $(1+\varepsilon)^2 \|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 = 1$. With this relation, we can perform suitable choices of c and $1/c$ such that our estimate becomes

$$\begin{aligned} -2\left\langle Kx, y \right\rangle &\geq -\left(\frac{\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 \|x\|_X^2}{(1+\varepsilon)\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2} + \frac{\|y\|_Y^2}{(1+\varepsilon)} \right) \\ &\geq -\left(\frac{1}{1+\varepsilon} \right) \left(\|x\|_X^2 + \|y\|_Y^2 \right) \;. \end{aligned}$$

Inserting this estimate into (9), we obtain $(\frac{\varepsilon}{1+\varepsilon})(\|x\|_X^2 + \|y\|_Y^2) > 0$. Since $\varepsilon > 0$, it becomes clear that the condition (7) ensures positive definiteness of M. □

Remark 1. Choosing $\mathrm{T} = \tau I$ and $\Sigma = \sigma I$, for some $\tau, \sigma > 0$, the assertion (7) reduces to $\tau\sigma\|K\|^2 < 1$, which is the original condition on the step widths τ and σ to ensure convergence of the primal-dual algorithm [5]. Note that this criterion requires to compute (or to estimate) the operator norm of K which can be difficult in some situations.

Theorem 1. *Let $\theta = 1$ and T, Σ be symmetric and positive definite matrices such that $\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 < 1$. Then, the sequence (x^n, y^n) generated by the iterates* (4) *converges weakly to an optimal solution (x^*, y^*) of the saddle-point problem* (1).

Proof. As shown in Lemma 1, the conditions $\theta = 1$ and $\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 < 1$ ensure that the matrix M defined in (6) is symmetric and positive definite. Hence, the variational inequality (5) takes the form of a proximal point algorithm. Convergence of the iterates (4) to an optimal solution (x^*, y^*) of (1) then follows from the weak convergence of the proximal point algorithm (Theorem 1 of [14]). □

In the next Section we study simple diagonal preconditioners which obey (7) and hence ensure the convergence of the algorithm. Furthermore, computing these preconditioners only requires to compute row and column wise norms of K, which can be done very efficiently.

2.2. A family of diagonal preconditioners

In general, Σ and T could be any symmetric, positive definite maps. However, since it is a fundamental requirement of the proposed algorithm, that the resolvent operators are simple, some care has to be taken in choosing T and Σ. In particular, if G and F^* are separable in x_j and y_i, the resolvent operators remain simple, if Σ and T are restricted to diagonal matrices. In the following we study a family of preconditioners which satisfy all these requirements and additionally guarantee the convergence of the algorithm.

Lemma 2. *Let $\mathrm{T} = \mathrm{diag}(\boldsymbol{\tau})$, where $\boldsymbol{\tau} = (\tau_1, ...\tau_n)$ and $\Sigma = \mathrm{diag}(\boldsymbol{\sigma})$, where $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_m)$. In particular,*

$$\tau_j = \frac{1}{\sum_{i=1}^m |K_{i,j}|^{2-\alpha}} \;, \quad \sigma_i = \frac{1}{\sum_{j=1}^n |K_{i,j}|^{\alpha}} \tag{10}$$

then for any $\alpha \in [0, 2]$

$$\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 = \sup_{x \in X,\, x \neq 0} \frac{\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} x\|^2}{\|x\|^2} \leq 1 \;. \tag{11}$$

Proof. In order to prove the inequality, we need to find an upper bound on $\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} x\|^2$. We have

$$\begin{aligned} \|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} x\|^2 &= \left\langle \Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} x, \Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} x \right\rangle \\ &= \sum_{i=1}^m \left(\sum_{j=1}^n \sqrt{\sigma_i} K_{i,j} \sqrt{\tau_j} x_j \right)^2 \\ &= \sum_{i=1}^m \sigma_i \left(\sum_{j=1}^n K_{i,j} \sqrt{\tau_j} x_j \right)^2 \\ &\leq \sum_{i=1}^m \sigma_i \left(\sum_{j=1}^n |K_{i,j}|^{\frac{\alpha}{2}} |K_{i,j}|^{1-\frac{\alpha}{2}} \sqrt{\tau_j} x_j \right)^2 \end{aligned}$$

Then, by applying the Cauchy-Schwarz inequality we obtain

$$\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} x\|^2 \leq \sum_{i=1}^m \sigma_i \left(\sum_{j=1}^n |K_{i,j}|^{\alpha} \right) \left(\sum_{j=1}^n |K_{i,j}|^{2-\alpha} \tau_j x_j^2 \right)$$

By definition of σ_i and τ_j, the above inequality can be simplified to

$$\begin{aligned}\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} x\|^2 &\leq \sum_{i=1}^{m}\sum_{j=1}^{n} |K_{i,j}|^{2-\alpha} \tau_j x_j^2 \\ &= \sum_{j=1}^{n}\left(\sum_{i=1}^{m} |K_{i,j}|^{2-\alpha}\right)\tau_j x_j^2 = \|x\|^2 \ .\end{aligned}$$

Substituting back into the definition of the operator norm, we finally obtain

$$\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}}\|^2 = \sup_{x\in X,\, x\neq 0} \frac{\|\Sigma^{\frac{1}{2}} K \mathrm{T}^{\frac{1}{2}} x\|^2}{\|x\|^2} \leq \frac{\|x\|^2}{\|x\|^2} = 1 \tag{12}$$

□

Remark 2. To ensure positive definiteness in (8) inequality (11) has to be strict. This requirement can be easily satisfied by multiplying $\boldsymbol{\tau}$ and $\boldsymbol{\sigma}$ by appropriate positive constants μ and ν such that $\mu\nu < 1$. However, in practice, we did not observe any convergence problems of the algorithm for $\mu = \nu = 1$.

Remark 3. Basically, the diagonal preconditioning leads to dimension-dependent times steps τ_j and σ_i instead of global steps τ and σ of the algorithm in [5] and hence do not change the computational complexity of the algorithm.

2.3. A remark on the alternating step method

A monotropic program is defined as a convex optimization problem of the form

$$\min_x h(x)\ ,\ \text{s.t. } Ax = b\ , \tag{13}$$

where $x \in \mathbb{R}^n$, $h(x) = \sum_{j=1}^n h_j(x_j)$ with $h_j(\cdot)$ being convex functions, $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$. Let us denote by a_j the j^{th} column of A, by $r_i(x)$ the i^{th} entry of the vector $r(x) = b - Ax$ and by q_i the number of non-zero entries in the i^{th} row of A.

In [7], the alternating step method for monotropic programming has been introduced as a special case of the alternating direction method of multipliers. It is defined as the iterative scheme

$$\begin{cases} x_j^{k+1} = \arg\min_{x_j}\Big\{h_j(x_j) + \left\langle a_j, \pi^k\right\rangle x_j + \\ \qquad \frac{\lambda\|a_j\|^2}{2}\left(x_j - \left[x_j^k + \frac{1}{\|a_j\|^2}\sum_{i=1}^m \frac{a_{i,j} r_i(x^k)}{q_i}\right]\right)^2\Big\} \\ \pi_i^{k+1} = \pi_i^k + \frac{\lambda}{q_i} r_i(x^{k+1}) \end{cases} \tag{14}$$

where $\pi \in \mathbb{R}^m$ is the dual variable and $\lambda > 0$ is a parameter which weights between the strengths of the primal and dual updates.

Let us now cast (13) in the form of (1). Indeed, it can be written very easily as a saddle-point problem by introducing Lagrange multipliers y for the constraint $Ax = b$, giving us

$$\min_x \max_y \ \langle b - Ax, y\rangle + h(x). \tag{15}$$

The iterates (4) now become

$$\begin{cases} x^{k+1} = (I + \mathrm{T}\partial h)^{-1}(x^k + \mathrm{T}A^T y^k) \\ y^{k+1} = y^k + \Sigma(b - A(2x^{k+1} - x^k)) \end{cases} \tag{16}$$

Let us now perform the particular choice $\alpha = 0$ in (10) to compute T, Σ and perform an additional scaling by the factors $1/\lambda$ and λ, respectively (see Remark 2). We consequently find that

$$\tau_j = \frac{1}{\lambda\|a_j\|^2} \quad \text{and} \quad \sigma_i = \frac{\lambda}{q_i}\ ,$$

where we used the convention that $0^0 = 0$. By a change of variables in the duals, $\pi^k = y^k - \Sigma(b - Ax^k)$, it can be checked that in this particular setting, (16) is equivalent to the alternating step method (14).

3. Numerical results

In this Section we perform several numerical experiments using the proposed algorithm. We first show applications to general linear programming problems and show that the proposed algorithm leads to a significantly faster convergence compared to the algorithm in [5]. Furthermore, we show that for large scale problems, a simple Matlab implementation of the proposed algorithm significantly outperforms a highly optimized interior point solver of Matlab. Finally, we show applications of the proposed algorithm to classical computer vision problems such as graph cuts and minimal partitioning problems, where we show that a GPU implementation of the proposed algorithm can easily compete with specialized algorithms such as the maxflow algorithm of Boykov and Kolmogorov [3].

In our numerical experiments, we use the following algorithms and parameter settings:

- IP: Matlab's LP interior point solver `linprog` with settings optimized for large scale problems.
- PD: The primal-dual algorithm (2) where we set $\tau = \frac{1}{L}$ and $\sigma = \frac{1}{L}$ and $L = \|K\|$ is estimated using the Matlab command `normest`.
- P-PD: Preconditioned primal dual algorithm (4) with the preconditioners T and Σ defined in (10) and using $\alpha = 1$.

In general, the solutions of the problems we consider here are not unique. Therefore, we use the value of the objective functions as a criterion to determine the convergence of the algorithms. We define the relative error of the objective function as $\delta^k = |E^* - E^k|/|E^*|$, where E^* refers to the optimal value and E^k is the value of the current iterate. Depending of the particular problem, E might be the value of either the primal or the dual problem. In order to determine the optimal value of the objective function we run one method with a very high accuracy. During the evaluation, we run the algorithms until δ^k is below a fixed threshold of $\mathrm{tol} = 10^{-4}$. Note that for practical purposes, δ^k can also be bounded by the primal-dual gap. All experiments were executed on a 1.73 GHz i7 CPU running Matlab and a NVidia GTX480 GPU using the CUDA framework.

3.1. Linear programming

Linear programs (LPs) define an important class of optimization problems in computer vision. See for example the recent work [15] for a (huge scale) LP formulation of curvature dependent segmentation and inpainting models. Therefore, there is a strong demand for efficient solver which can handle very large problems.

Here, we consider LPs in so-called equality form:

$$\min\, c^T x \quad \text{s.t.} \quad Ax = b\,,\ x \geq 0\ , \tag{17}$$

where $x, c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times n}$. Note that it is straight-forward to transform other standard forms to this form [2]. The above LP can be easily reformulated in terms of a saddle-point problem by introducing Lagrange multipliers y to account for the equality constraints:

$$\min_x \max_y\, \langle Ax - b, y\rangle + \langle c, x\rangle\ ,\ \text{s.t. } x \geq 0$$

Applying P-PD to (17) we obtain the following simple iterates

$$\begin{cases} x^{k+1} = \mathrm{proj}_{[0,\infty)}\left(x^k - \mathrm{T}(A^T y^k + c)\right) \\ y^{k+1} = y^k + \Sigma(A(2x^{k+1} - x^k) - b)\ , \end{cases} \tag{18}$$

where $\mathrm{proj}_{[0,\infty)}(\cdot)$ can be implemented by simple truncation operations.

In our first example, we compare IP, PD and P-PD on two standard LP test problems that come along with the Matlab package. To gain more insight to the proposed preconditioning technique, we compare the global steps of PD with the dimension-dependent adaptive steps of P-PD.

- `sc50b`: This problem consists of 48 variables, 30 inequalities and 20 equalities. We find $L = 5.3871$ and set $\tau = \sigma = 0.1856$ for PD. For P-PD we find that the average primal step is $\bar{\tau} = 0.3387$ and the average dual step is $\bar{\sigma} = 0.3479$. Therefore, P-PD can take in average approximately 1.8 times larger primal and dual steps than PD.
- `densecolumn`: This problem consists of 1677 variables and 627 equalities with one dense column in the linear operator. We find $L = 9.6817 \cdot 10^3$ and consequently set $\tau = \sigma = 1.0329 \cdot 10^{-4}$ for PD. For P-PD, we find the average primal and dual steps to be $\bar{\tau} = 0.0044$ and $\bar{\sigma} = 0.0016$. Hence P-PD can take on average approximately 40 times larger primal and 15 times larger dual steps.

Table 1 shows the results of the performance evaluation on the standard LP test problems. The optimal value of the objective function was determined by running IP with a very high accuracy. This examples clearly demonstrate the advantage of the proposed algorithm. On `sc50b` P-PD is 1.4 times faster than PD while on `densecolumn`, P-PD is more than 200 times faster than PD. Especially, `densecolumn` has a very irregular structure which can be handled much better by the preconditioned algorithm. Clearly, on small-scale problems, the simple primal-dual algorithms can not compete with the highly optimized IP. However, as we will see in the next section, this is quite different for large-scale problems. Figure 1 depicts convergence plots for PD and P-PD for the two LP test problems.

	IP	PD	P-PD
`sc50b`	0.01s	1.75s	0.49s
`densecolumn`	0.17s	268.51s	0.61s

Table 1. Comparison of of IP, PD and P-PD on two standard LP test problems.

3.2. TV-L^1 image denoising

The TV-L^1 model [12, 6] is a popular image denoising model. It consists of total variation (TV) regularization and a L^1 data fidelity term. It is known that the TV-L^1 model is very effective in removing strong outliers such as salt and pepper noise in images. We assume discrete images defined on a regular Cartesian grid of size $M \times N$ with indices (i, j). The (anisotropic) TV-L^1 model in a discrete setting is defined as the ℓ_1 minimization problem

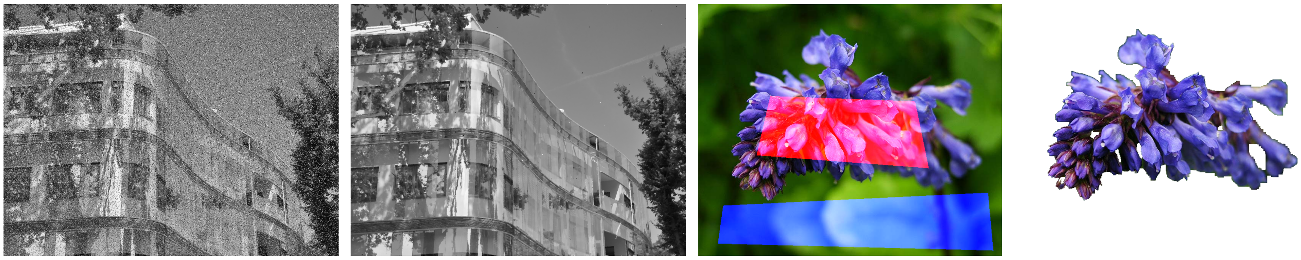
$$\min_u \|Du\|_{\ell_1} + \lambda \|u - f\|_{\ell_1}\ , \tag{19}$$

where $f = (f_{i,j}) \in \mathbb{R}^{MN}$ is the noisy input image and $u = (u_{i,j}) \in \mathbb{R}^{MN}$ is the denoised image. The free parameter $\lambda > 0$ is used to control the trade-off between smoothing and data fidelity. The linear operator $D \in \mathbb{R}^{2MN \times MN}$ is the discretized gradient operator defined by

$$(Du)_{i,j} = \begin{pmatrix} (Du)^1_{i,j} \\ (Du)^2_{i,j} \end{pmatrix}\ , \tag{20}$$

where

$$(Du)^1_{i,j} = \begin{cases} u_{i+1,j} - u_{i,j} & \text{if } i < M \\ 0 & \text{if } i = M \end{cases}\ ,$$



(a) Noisy image (b) Denoised image (c) Input image (d) Graph cut

Figure 2. Figures (a-b) show the result of TV-L^1 denoising of a image containing 15% salt and pepper noise. Figures (c-d) show the result of binary graph cut segmentation.

and

$$(Du)^2_{i,j} = \begin{cases} u_{i,j+1} - u_{i,j} & \text{if } j < N \\ 0 & \text{if } j = N \end{cases} .$$

First, we convert (19) to a saddle-point problem by dualizing both ℓ_1 norms.

$$\min_u \max_{p,q} \langle Du, p\rangle + \lambda \langle u - f, q\rangle \quad \text{s.t.} \quad -1 \leq (p,q) \leq 1 \, ,$$

where $p = ((p^1_{i,j}, p^2_{i,j})) \in \mathbb{R}^{2MN}$ and $q = (q_{i,j})$ are the dual variables. Then, by pointwise minimizing the above problem in u, we end up with the following LP formulation, which is the dual of (19):

$$\max_q -f^T q \quad \text{s.t. } D^T p + \text{diag}(\lambda I) q = 0 \, , \ -1 \leq (p,q) \leq 1 \, .$$

This LP can be solved using a slight variant of (18). After solving the dual problem, the solution u of the primal problem can be recovered from the dual variables of the primal-dual algorithm.

Figure 2 shows the result of applying the TV-L^1 model to a noisy input image. The smoothing parameter was set to $\lambda = 2$. Solving the TV-L^1 model for this problem means to compute $1024 \times 768 = 786\,432$ unknowns and hence the associated LP problem is clearly of large scale. We ran PD, P-PD and IP until the relative error δ^k was below tol. Table 2 shows the results of the comparison. Note that a simple Matlab implementation of P-PD is already more than 80 times faster that IP.

IP	PD	P-PD
1784.53s	24.29s	21.12s

Table 2. Comparison of the proposed algorithm for TV-L^1 denoising.

3.3. Graph cuts

Graph cuts belong to one of the most successful algorithms in computer vision. Their major aim is to partition a given graph into a disjoint set of nodes, such that the sum of the weights of the edges that connect these sets yield a minimum value. It is well known that the minimum cut in a graph can be computed by finding the maximum flow from the source to the sink. The well-known Ford and Fulkerson theorem [9] states that the maximum flow will saturate a set of edges that will partition the graph into two disjoint sets connected to the source and sink. Furthermore, the value of the maximum flow is equal to the value of the minimum cut. A very efficient combinatorial algorithm to compute the solution of the maximum flow problem has been proposed by Boykov and Kolmogorov in [3].

Let us consider a four-connected graph representing a regular Cartesian grid of size $M \times N$ with indices (i,j). Let $u = (u_{i,j}) \in \mathbb{R}^{MN}$ be the labeling function, $w^u = (w^u_{i,j}) \in \mathbb{R}^{MN}$ be the unary weights and $w^b = ((w^{b,1}_{i,j}, w^{b,2}_{i,j})) \in (\mathbb{R}^+)^{2MN}$ be the binary weights. The minimum cut problem can be written (see for instance [4]) in terms of the following weighted TV energy

$$\min_u \|D_w u\|_{\ell_1} + \langle u, w^u \rangle \, , \text{ s.t. } 0 \leq u \leq 1 \, , \tag{21}$$

where $D_w = \text{diag}(w^b)D$ is the weighted gradient operator. This formulation is closely related to [1] but a bit more efficient since we do not need extra dual variables for the unary term.

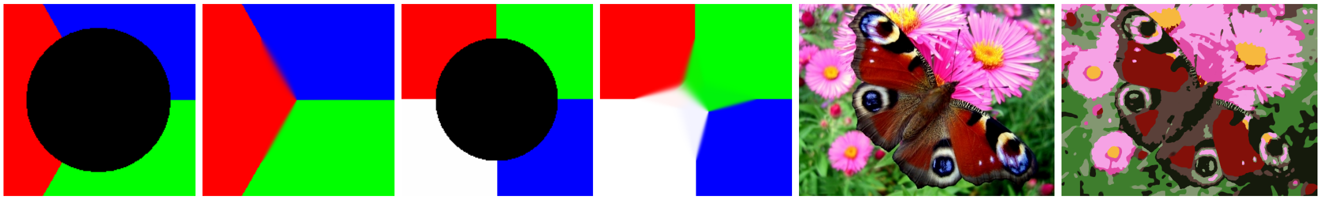
Dualizing the ℓ_1 norm in (21), we arrive at the following saddle point problem

$$\begin{aligned} \min_u \max_p \quad & \langle D_w u, p\rangle + \langle u, w^u\rangle \\ \text{s.t.} \quad & 0 \leq u \leq 1 \, , \quad -1 \leq p \leq 1 \end{aligned}$$

where $p = (p^1_{i,j}, p^2_{i,j}) \in \mathbb{R}^{2MN}$ are the dual variables. Applying algorithm (4) to the above problem, we end up with the following simple iterates

$$\begin{cases} u^{k+1} = \text{proj}_{[0,1]} \left(u^k + \mathrm{T}(D_w^T p^k - w^u) \right) \\ p^{k+1} = \text{proj}_{[-1,1]} \left(p^k + \Sigma(D_w(2u^{k+1} - u^k)) \right) \end{cases} , \tag{22}$$

where $\text{proj}_{[0,1]}(\cdot)$ and $\text{proj}_{[-1,1]}(\cdot)$ can be implemented by simple pointwise truncation operations. Note that the proposed algorithm can be easily modified for minimizing the



(a) 3 phases (b) Output (c) 4 phases (d) Output (e) Natural (f) Segmentation (8 phases)

Figure 3. Figures (a-b) and (c-d) show the result of standard Dirichlet problems. Figures (e-f) show the segmentation of a natural image into 8 regions.

usual isotropic total variation. The only modification is to replace the projections of the duals onto box constraints by projections onto ℓ_2 balls.

Figure 2 shows the result of PD and P-PD applied to a binary image segmentation problem. Let $I \in [0,1]^{3MN}$ be the input image. The unary terms were computed by first computing the average RGB values $\mu_f \in \mathbb{R}^3$ and $\mu_b \in \mathbb{R}^3$ in the user specified foreground and background regions and then by setting $w^u_{i,j} = \alpha(\|I_{i,j} - \mu_f\|^2 - \|I_{i,j} - \mu_b\|^2)$. The binary terms were computed by $w^{b,1}_{i,j} = \exp(-\beta|I_{i+1,j} - I_{i,j}|)$ and $w^{b,2}_{i,j} = \exp(-\beta|I_{i,j+1} - I_{i,j}|)$, where we set $\alpha = 1$ and $\beta = 20$.

We used the popular maximum flow algorithm of Boykov and Kolmogorov [3] to compute the optimal value of the primal objective function (21). We then ran PD and P-PD until the relative error δ^k was below tol. Table 3 presents the timings. Although a Matlab implementation of P-PD cannot compete with the highly optimized maximum flow algorithm, a parallel GPU implementation is already significantly faster.

MAXFLOW	PD	P-PD	P-PD-GPU
0.160s	15.75s	8.56s	0.045s

Table 3. Comparison of the proposed algorithm for graph cut segmentation.

3.4. Minimal partitions

In the last example we quickly explain the application of the proposed preconditioned primal-dual algorithm for computing minimal partitions. Following [5] and references therein, a convex relaxation of the multi-label Potts model can be computed by solving the saddle point problem

$$\min_{u \in S} \max_{q \in B} \sum_{l=1}^{k} \langle Du_l, q_l \rangle + \langle u_l, f_l \rangle \ , \qquad (23)$$

where

$$S = \left\{ u = (u_l)_{l=1}^k : \sum_{l=1}^{k} u_l = 1 \, , \ u_l \geq 0 \, , \right\} \qquad (24)$$

is the usual simplex constraint and the convex set

$$B = \bigcap_{1 \leq m < n \leq k} B_m^n \ , \qquad (25)$$

can be written as the intersection of balls of the form

$$B_m^n = \left\{ q = ((q_l^1), (q_l^2))_{l=1}^k : (|q_m - q_n|)_{i,j} \leq 1 \, , \ \forall \, i,j \right\} \ , \qquad (26)$$

The linear operator D is the discretized gradient operator as defined in (20), $u = (u_l)_{l=1}^k$, is the labeling function which represents a relaxed assignment of the image pixels to each region, $f = (f_l)_{l=1}^k$ is some given weight function and $q = ((q_l^1), (q_l^2))_{l=1}^k$ are the dual variables.

The original approach taken in [5] is to compute the projections with respect to S and B by solving auxiliary optimization problems. This approach has the disadvantage that the inner optimization problems have to be computed with sufficient accuracy such that the overall algorithm converges, which can heavily slow down the algorithm.

Here, we take a different approach, which is closely related to the multiple operator splitting method proposed in [11]. We first introduce slack variables b for the expressions $q_m - q_n$ in (26). Then we introduce Lagrange multipliers λ to account for the equality constraints $\sum_l u_l = 1$ in (24) and μ to account for the equality constraints $b_m^n = q_m - q_n$. Together, this yields the following augmented saddle-point problem

$$\begin{aligned} \min_{u,\mu} \max_{q,b,\lambda} \quad & \textstyle\sum_{k=1}^{l} \langle Du_l, q_l \rangle + \langle u_l, f_l \rangle \\ & + \langle Pu - 1, \lambda \rangle + \langle b - Qq, \mu \rangle \\ \text{s.t.} \quad & (u_l)_{i,j} \geq 0 \, , \ |(b_m^n)_{i,j}| \leq 1 \, , \end{aligned} \qquad (27)$$

where the linear operators P and Q are such that

$$(Pu)_{i,j} = \sum_{l=1}^{k} (u_l)_{i,j} \, , \quad ((Qq)_m^n)_{i,j} = (q_m - q_n)_{i,j} \ .$$

Note that (27) now depends only on simple pointwise constraints for which the projections can be implemented very efficiently. To cast the above problem into the form of (1), it remains to arrange all primal variables in a vector x, all dual variables into a vector y and all linear operators into a global linear operator K. Then, applying the preconditioning techniques proposed in this paper leads to an algorithm that is guaranteed to converge to the optimal solution without the need to solve any inner optimization problems.

Figure 3 shows some results of standard minimal partitioning and segmentation problems. We compared the original approach solving inner optimization problems and using PD to P-PD applied to (27). We first precomputed the optimal solution using a large number of iterations and then recorded the time until the error is below a threshold of tol. The timings are presented in Table 4. In all cases, the proposed algorithm clearly outperforms the original approach of [5].

	PD	P-PD	Speedup
Synthetic (3 phases)	221.71s	75.65s	2.9
Synthetic (4 phases)	1392.02s	538.83s	2.6
Natural (8 phases)	592.85s	113.76s	5.2

Table 4. Comparison of the proposed algorithm on partitioning problems.

4. Conclusion

In this paper we have proposed a simple preconditioning technique to improve the performance of the first-order primal-dual algorithm proposed in [13, 5]. The proposed diagonal preconditioners can be computed efficiently and guarantee the convergence of the algorithm without the need to estimate any step size parameters. In several numerical experiments, we have shown that the proposed algorithm significantly outperforms the algorithm in [5]. Furthermore, on large scale linear programming problems, an unoptimized implementation of the proposed algorithm easily outperforms a highly optimized interior point solver and a GPU implementation of the proposed algorithm can easily compete with specialized combinatorial algorithms for computing minimum cuts.

We believe that the proposed algorithm can become a standard algorithm in computer vision since it can be applied to a large class of convex optimization problems arising in computer vision and has the potential for parallel computing. Future work will mainly concentrate on the development of more sophisticated preconditioners that are different from diagonal matrices.

References

[1] A. Bhusnurmath and C. Taylor. Graph cuts via ℓ_1 norm minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1866–1871, 2008.

[2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[3] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in computer vision. In A. J. M. Figueiredo, J. Zerubia, editor, *Energy Minimization Methods in Computer Vision and Pattern Recognition*, volume 2134 of *LNCS*, pages 359–374. Springer, 2001.

[4] A. Chambolle. Total variation minimization and a class of binary MRF models. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 136–152, 2005.

[5] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 2010. online first.

[6] T. Chan and S. Esedoglu. Aspects of total variation regularized L^1 function approximation. *SIAM J. Appl. Math.*, 65(5):1817–1837, 2004.

[7] J. Eckstein. *Splitting Methods for Monotone Operators with Application to Parallel Optimization*. PhD thesis, MIT, 1989.

[8] E. Esser, X. Zhang, and T. Chan. A general framework for a class of first order primal-dual algorithms for convex optimization in imaging science. *SIAM J. Imaging Sci.*, 3:1015–1046, 2010.

[9] L. Ford and D. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, New Jersey, 1962.

[10] B. He and X. Yuan. Convergence analysis of primal-dual algorithms for total variation image restoration. Technical report, Nanjing University, China, 2010.

[11] J. Lellmann, D. Breitenreicher, and C. Schnörr. Fast and exact primal-dual iterations for variational problems in computer vision. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *European Conference on Computer Vision (ECCV)*, volume 6312 of *LNCS*, pages 494–505. Springer Berlin / Heidelberg, 2010.

[12] M. Nikolova. A variational approach to remove outliers and impulse noise. *Journal of Mathematical Imaging and Vision*, 20(1-2):99–120, 2004.

[13] T. Pock, D. Cremers, H. Bischof, and A. Chambolle. An algorithm for minimizing the Mumford-Shah functional. In *International Conference on Computer Vision (ICCV)*, 2009.

[14] R. T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14(5):877–898, 1976.

[15] T. Schoenemann, F. Kahl, and D. Cremers. Curvature regularity for region-based image segmentation and inpainting: A linear programming relaxation. In *IEEE International Conference on Computer Vision (ICCV)*, Kyoto, Japan, 2009.

[16] X. Zhang, M. Burger, and S. Osher. A unified primal-dual algorithm framework based onbregman iteration. *Journal of Scientific Computing*, 46:20–46, 2011.