An ADMM Algorithm for a Class of Total Variation Regularized Estimation Problems ⋆

Bo Wahlberg *, **Stephen Boyd** **, **Mariette Annergren** *, **and Yang Wang** **

* *Automatic Control Lab and ACCESS, School of Electrical Engineering, KTH Royal Institute of Technology, SE 100 44 Stockholm,Sweden*
** *Department of Electrical Engineering, Stanford University, Stanford, CA 94305, USA*

Abstract: We present an alternating augmented Lagrangian method for convex optimization problems where the cost function is the sum of two terms, one that is separable in the variable blocks, and a second that is separable in the difference between consecutive variable blocks. Examples of such problems include Fused Lasso estimation, total variation denoising, and multi-period portfolio optimization with transaction costs. In each iteration of our method, the first step involves separately optimizing over each variable block, which can be carried out in parallel. The second step is not separable in the variables, but can be carried out very efficiently. We apply the algorithm to segmentation of data based on changes in mean (ℓ_1 mean filtering) or changes in variance (ℓ_1 variance filtering). In a numerical example, we show that our implementation is around 10000 times faster compared with the generic optimization solver SDPT3.

Keywords: Signal processing algorithms, stochastic parameters, parameter estimation, convex optimization and regularization

1. INTRODUCTION

In this paper we consider optimization problems where the objective is a sum of two terms: The first term is separable in the variable blocks, and the second term is separable in the difference between consecutive variable blocks. One example is the Fused Lasso method in statistical learning, Tibshirani et al. [2005], where the objective includes an ℓ_1-norm penalty on the parameters, as well as an ℓ_1-norm penalty on the difference between consecutive parameters. The first penalty encourages a sparse solution, *i.e.*, one with few nonzero entries, while the second penalty enhances block partitions in the parameter space. The same ideas have been applied in many other areas, such as Total Variation (TV) denoising, Rudin et al. [1992], and segmentation of ARX models, Ohlsson et al. [2010] (where it is called sum-of-norms regularization). Another example is multi-period portfolio optimization, where the variable blocks give the portfolio in different time periods, the first term is the portfolio objective (such as risk-adjusted return), and the second term accounts for transaction costs.

In many applications, the optimization problem involves a large number of variables, and cannot be efficiently handled by generic optimization solvers. In this paper, our main contribution is to derive an efficient and scalable optimization algorithm, by exploiting the structure of the optimization problem. To do this, we use a distributed optimization method called Alternating Direction Method of Multipliers (ADMM). ADMM was developed in the 1970s, and is closely related to many other optimization algorithms including Bregman iterative algorithms for ℓ_1 problems, Douglas-Rachford splitting, and proximal point methods; see Eckstein and Bertsekas [1992], Combettes and Pesquet [2007]. ADMM has been applied in many areas, including image and signal processing, Setzer [2011], as well as large-scale problems in statistics and machine learning, Boyd et al. [2011].

We will apply ADMM to ℓ_1 mean filtering and ℓ_1 variance filtering (Wahlberg et al. [2011]), which are important problems in signal processing with many applications, for example in financial or biological data analysis. In some applications, mean and variance filtering are used to pre-process data before fitting a parametric model. For non-stationary data it is also important for segmenting the data into stationary subsets. The approach we present is inspired by the ℓ_1 trend filtering method described in Kim et al. [2009], which tracks changes in the mean value of the data. (An example in this paper also tracks changes in the variance of the underlying stochastic process.) These problems are closely related to the covariance selection problem, Dempster [1972], which is a convex optimization problem when the inverse covariance is used as the optimization variable, Banerjee et al. [2008]. The same ideas can also be found in Kim et al. [2009] and Friedman et al. [2008].

⋆ This work was partially supported by the Swedish Research Council, the Linnaeus Center ACCESS at KTH and the European Research Council under the advanced grant LEARN, contract 267381.

This paper is organized as follows. In Section 2 we review the ADMM method. In Section 3, we apply ADMM to our optimization problem to derive an efficient optimization algorithm. In Section 4.1 we apply our method to ℓ_1 mean filtering, while in Section 4.2 we consider ℓ_1 variance filtering. Section 5 contains some numerical examples, and Section 6 concludes the paper.

2. ALTERNATING DIRECTION METHOD OF MULTIPLIERS (ADMM)

In this section we give an overview of ADMM. We follow closely the development in Section 5 of Boyd et al. [2011].

Consider the following optimization problem

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \mathcal{C} \end{array} \tag{1}$$

with variable $x \in \mathbb{R}^n$, and where f and $\mathcal{C}$ are convex. We let $p^\star$ denote the optimal value of (1). We first re-write the problem as

$$\begin{array}{ll} \text{minimize} & f(x) + I_{\mathcal{C}}(z) \\ \text{subject to} & x = z, \end{array} \tag{2}$$

where $I_{\mathcal{C}}(z)$ is the indicator function on $\mathcal{C}$ (*i.e.*, $I_{\mathcal{C}}(z) = 0$ for $z \in \mathcal{C}$, and $I_{\mathcal{C}}(z) = \infty$ for $z \notin \mathcal{C}$). The augmented Lagrangian for this problem is

$$L_\rho(x, z, u) = f(x) + I_{\mathcal{C}}(z) + (\rho/2)\|x - z + u\|_2^2,$$

where u is a scaled dual variable associated with the constraint $x = z$, *i.e.*, $u = (1/\rho)y$, where y is the dual variable for $x = z$. Here, $\rho > 0$ is a penalty parameter.

In each iteration of ADMM, we perform alternating minimization of the augmented Lagrangian over x and z. At iteration k we carry out the following steps

$$x^{k+1} := \operatorname*{argmin}_x \{f(x) + (\rho/2)\|x - z^k + u^k\|_2^2\} \tag{3}$$

$$z^{k+1} := \Pi_{\mathcal{C}}(x^{k+1} + u^k) \tag{4}$$

$$u^{k+1} := u^k + (x^{k+1} - z^{k+1}), \tag{5}$$

where $\Pi_{\mathcal{C}}$ denotes Euclidean projection onto $\mathcal{C}$. In the first step of ADMM, we fix z and u and minimize the augmented Lagrangian over x; next, we fix x and u and minimize over z; finally, we update the dual variable u.

2.1 Convergence

Under mild assumptions on f and $\mathcal{C}$, we can show that the iterates of ADMM converge to a solution; specifically, we have

$$f(x^k) \to p^\star, \quad x^k - z^k \to 0,$$

as $k \to \infty$. The rate of convergence, and hence the number of iterations required to achieve a specified accuracy, can depend strongly on the choice of the parameter ρ. When ρ is well chosen, this method can converge to a fairly accurate solution (good enough for many applications), within a few tens of iterations. However, if the choice of ρ is poor, many iterations can be needed for convergence. These issues, including heuristics for choosing ρ, are discussed in more detail in Boyd et al. [2011].

2.2 Stopping criterion

The primal and dual residuals at iteration k are given by

$$e_p^k = (x^k - z^k), \quad e_d^k = -\rho(z^k - z^{k-1}).$$

We terminate the algorithm when the primal and dual residuals satisfy a stopping criterion (which can vary depending on the requirements of the application). A typical criterion is to stop when

$$\|e_p^k\|_2 \le \epsilon^{\text{pri}}, \quad \|e_d^k\|_2 \le \epsilon^{\text{dual}}.$$

Here, the tolerances $\epsilon^{\text{pri}} > 0$ and $\epsilon^{\text{dual}} > 0$ can be set via an absolute plus relative criterion,

$$\epsilon^{\text{pri}} = \sqrt{n}\epsilon^{\text{abs}} + \epsilon^{\text{rel}} \max\{\|x^k\|_2, \|z^k\|_2\},$$
$$\epsilon^{\text{dual}} = \sqrt{n}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}\rho\|u^k\|_2,$$

where $\epsilon^{\text{abs}} > 0$ and $\epsilon^{\text{rel}} > 0$ are absolute and relative tolerances (see Boyd et al. [2011] for details).

3. PROBLEM FORMULATION AND METHOD

In this section we formulate our problem and derive an efficient distributed optimization algorithm via ADMM.

3.1 Optimization problem

We consider the problem

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^{N} \Phi_i(x_i) + \sum_{i=1}^{N-1} \Psi_i(r_i) \\ \text{subject to} & r_i = x_{i+1} - x_i, \quad i = 1, \ldots, N-1 \end{array} \tag{6}$$

with variables $x_1, \ldots, x_N, r_1, \ldots, r_{N-1} \in \mathbf{R}^n$, and where $\Phi_i : \mathbf{R}^n \to \mathbf{R} \cup \{\infty\}$ and $\Psi_i : \mathbf{R}^n \to \mathbf{R} \cup \{\infty\}$ are convex functions.

This problem has the form (1), with variables $x = (x_1, \ldots, x_N)$, $r = (r_1, \ldots, r_{N-1})$, objective function

$$f(x, r) = \sum_{i=1}^{N} \Phi_i(x_i) + \sum_{i=1}^{N-1} \Psi_i(r_i)$$

and constraint set

$$\mathcal{C} = \{(x, r) \mid r_i = x_{i+1} - x_i,\ i = 1, \ldots, N-1\}. \tag{7}$$

The ADMM form for problem (6) is

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^{N} \Phi_i(x_i) + \sum_{i=1}^{N-1} \Psi_i(r_i) + I_{\mathcal{C}}(z, s) \\ \text{subject to} & r_i = s_i, \quad i = 1, \ldots, N-1 \\ & x_i = z_i, \quad i = 1, \ldots, N, \end{array} \tag{8}$$

with variables $x = (x_1, \ldots, x_N)$, $r = (r_1, \ldots, r_{N-1})$, $z = (z_1, \ldots, z_N)$, and $s = (s_1, \ldots, s_{N-1})$. Furthermore, we let $u = (u_1, \ldots, u_N)$ and $t = (t_1, \ldots, t_{N-1})$ be vectors of scaled dual variables associated with the constraints $x_i = z_i$, $i = 1, \ldots, N$, and $r_i = s_i$, $i = 1, \ldots, N-1$ (*i.e.*, $u_i = (1/\rho)y_i$, where y_i is the dual variable associated with $x_i = z_i$).

3.2 Distributed optimization method

Applying ADMM to problem (8), we carry out the following steps in each iteration.

Step 1. Since the objective function f is separable in x_i and r_i, the first step (3) of the ADMM algorithm consists of $2N - 1$ separate minimizations

$$x_i^{k+1} := \operatorname*{argmin}_{x_i} \{\Phi_i(x_i) + (\rho/2)\|x_i - z_i^k + u_i^k\|_2^2\}, \tag{9}$$

$i = 1, \ldots, N$, and

$$r_i^{k+1} := \underset{r_i}{\operatorname{argmin}}\{\Psi_i(r_i) + (\rho/2)\|r_i - s_i^k + t_i^k\|_2^2\}, \quad (10)$$

$i = 1, \ldots, N-1$. These updates can all be carried out in parallel. For many applications, we will see that we can often solve (9) and (10) analytically.

Step 2. In the second step of ADMM, we project $(x^{k+1} + u^k, r^{k+1} + t^k)$ onto the constraint set $\mathcal{C}$, *i.e.*,

$$(z^{k+1}, s^{k+1}) := \Pi_{\mathcal{C}}((x^{k+1}, r^{k+1}) + (u^k, t^k)).$$

For the particular constraint set (7), we will show in Section 3.3 that the projection can be performed extremely efficiently.

Step 3. Finally, we update the dual variables:

$$u_i^{k+1} := u_i^k + (x_i^{k+1} - z_i^{k+1}), \quad i = 1, \ldots, N$$

and

$$t_i^{k+1} := t_i^k + (r_i^{k+1} - s_i^{k+1}), \quad i = 1, \ldots, N-1.$$

These updates can also be carried out independently in parallel, for each variable block.

3.3 Projection

In this section we work out an efficient formula for projection onto the constraint set $\mathcal{C}$ (7). To perform the projection

$$(z, s) = \Pi_{\mathcal{C}}((w, v)),$$

we solve the optimization problem

$$\begin{array}{ll} \text{minimize} & \|z - w\|_2^2 + \|s - v\|_2^2 \\ \text{subject to} & s = Dz, \end{array}$$

with variables $z = (z_1, \ldots, z_N)$ and $s = (s_1, \ldots, s_{N-1})$, and where $D \in \mathbf{R}^{(N-1)n \times Nn}$ is the forward difference operator, *i.e.*,

$$D = \begin{bmatrix} -I & I & & \\ & -I & I & \\ & & \ddots & \ddots \\ & & & -I \;\; I \end{bmatrix}.$$

This problem is equivalent to

$$\text{minimize } \|z - w\|_2^2 + \|Dz - v\|_2^2.$$

with variable $z = (z_1, \ldots, z_N)$. Thus to perform the projection we first solve the optimality condition

$$(I + D^T D)z = w + D^T v, \quad (11)$$

for z, then we let $s = Dz$.

The matrix $I + D^T D$ is block tridiagonal, with diagonal blocks equal to multiples of I, and sub/super-diagonal blocks equal to $-I$. Let LL^T be the Cholesky factorization of $I + D^T D$. It is easy to show that L is block banded with the form

$$L = \begin{bmatrix} l_{1,1} & & & & \\ l_{2,1} & l_{2,2} & & & \\ & l_{3,2} & l_{3,3} & & \\ & & \ddots & \ddots & \\ & & & l_{N,N-1} & l_{N,N} \end{bmatrix} \otimes I,$$

where $\otimes$ denotes the Kronecker product. The coefficients $l_{i,j}$ can be explicitly computed via the recursion

$$\begin{array}{l} l_{1,1} = \sqrt{2}, \\ l_{i+1,i} = -1/l_{i,i}, \;\; l_{i+1,i+1} = \sqrt{3 - l_{i+1,i}^2}, \;\; i = 1, \ldots, N-2, \\ l_{N,N-1} = -1/l_{N-1,N-1}, \quad l_{N,N} = \sqrt{2 - l_{N,N-1}^2}. \end{array}$$

The coefficients only need to be computed once, before the projection operator is applied.

The projection therefore consists of the following steps

(1) Form $b := w + D^T v$:

$$\begin{array}{l} b_1 := w_1 - v_1, \quad b_N := w_N + v_{N-1}, \\ b_i := w_i + (v_{i-1} - v_i), \quad i = 2, \ldots, N-1. \end{array}$$

(2) Solve $Ly = b$:

$$\begin{array}{l} y_1 := (1/l_{1,1})b_1, \\ y_i := (1/l_{i,i})(b_i - l_{i,i-1}y_{i-1}), \quad i = 2, \ldots, N. \end{array}$$

(3) Solve $L^T z = y$:

$$\begin{array}{l} z_N := (1/l_{N,N})y_N, \\ z_i := (1/l_{i,i})(y_i - l_{i+1,i}z_{i+1}), \quad i = N-1, \ldots, 1. \end{array}$$

(4) Set $s = Dz$:

$$s_i := z_{i+1} - z_i, \quad i = 1, \ldots, N-1.$$

Thus, we see that we can perform the projection very efficiently, in $\mathcal{O}(Nn)$ flops (floating-point operations). In fact, if we pre-compute the inverses $1/l_{i,i}$, $i = 1, \ldots, N$, the only operations that are required are multiplication, addition, and subtraction. We do not need to perform division, which can be expensive on some hardware platforms.

4. EXAMPLES

4.1 ℓ_1 Mean filtering

Consider a sequence of vector random variables

$$Y_i \sim \mathcal{N}(\bar{y}_i, \Sigma), \quad i = 1, \ldots, N,$$

where $\bar{y}_i \in \mathbf{R}^n$ is the mean, and $\Sigma \in \mathbf{S}_+^n$ is the covariance matrix. We assume that the covariance matrix is known, but the mean of the process is unknown. Given a sequence of observations $y_1, \ldots, y_N$, our goal is to estimate the mean under the assumption that it is piecewise constant, *i.e.*, $\bar{y}_{i+1} = \bar{y}_i$ for many values of i.

In the Fused Group Lasso method, we obtain our estimates by solving

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^{N} \frac{1}{2}(y_i - x_i)^T \Sigma^{-1}(y_i - x_i) + \lambda \sum_{i=1}^{N-1} \|r_i\|_2 \\ \text{subject to} & r_i = x_{i+1} - x_i, \quad i = 1, \ldots, N-1, \end{array}$$

with variables $x_1, \ldots, x_N$, $r_1, \ldots, r_{N-1}$. Let $x_1^\star, \ldots, x_N^\star$, $r_1^\star, \ldots, r_{N-1}^\star$ denote an optimal point, our estimates of $\bar{y}_1, \ldots, \bar{y}_N$ are $x_1^\star, \ldots, x_N^\star$.

This problem is clearly in the form (6), with

$$\Phi_i(x_i) = \frac{1}{2}(y_i - x_i)^T \Sigma^{-1}(y_i - x_i), \quad \Psi_i(r_i) = \lambda \|r_i\|_2.$$

ADMM steps. For this problem, steps (9) and (10) of ADMM can be further simplified. Step (9) involves mini-

mizing an unconstrained quadratic function in the variable x_i, and can be written as

$$x_i^{k+1} = (\Sigma^{-1} + \rho I)^{-1}(\Sigma^{-1} y_i + \rho(z_i^k - u_i^k)).$$

Step (10) is

$$r_i^{k+1} := \operatorname*{argmin}_{r_i}\{\lambda\|r_i\|_2 + (\rho/2)\|r_i - s_i^k + t_i^k\|_2^2\},$$

which simplifies to

$$r_i^{k+1} = \mathcal{S}_{\lambda/\rho}(s_i^k - t_i^k), \tag{12}$$

where $\mathcal{S}_\kappa$ is the vector soft thresholding operator, defined as

$$\mathcal{S}_\kappa(a) = (1 - \kappa/\|a\|_2)_+ a, \quad \mathcal{S}_\kappa(0) = 0.$$

Here the notation $(v)_+ = \max\{0, v\}$ denotes the positive part of the vector v. (For details see Boyd et al. [2011].)

Variations. In some problems, we might expect that individual components of x_t will be piecewise constant, in which case we can instead use the standard Fused Lasso method. In the standard Fused Lasso method we solve

$$\begin{array}{ll}\text{minimize} & \displaystyle\sum_{i=1}^{N} \frac{1}{2}(y_i - x_i)^T \Sigma^{-1}(y_i - x_i) + \lambda \sum_{i=1}^{N-1} \|r_i\|_1 \\ \text{subject to} & r_i = x_{i+1} - x_i, \quad i = 1, \ldots, N,\end{array}$$

with variables $x_1, \ldots, x_N$, $r_1, \ldots, r_{N-1}$. The ADMM updates are the same, except that instead of doing vector soft thresholding for step (10), we perform scalar componentwise soft thresholding, *i.e.*,

$$(r_i^{k+1})_j = \mathcal{S}_{\lambda/\rho}((s_i^k - t_i^k)_j), \quad j = 1, \ldots, n.$$

4.2 ℓ_1 Variance filtering

Consider a sequence of vector random variables (of dimension n)

$$Y_i \sim \mathcal{N}(0, \Sigma_i), \quad i = 1, \ldots, N,$$

where $\Sigma_i \in \mathbf{S}_+^n$ is the covariance matrix for Y_i (which we assume is fixed but unknown). Given observations of $y_1, \ldots, y_N$, our goal is to estimate the sequence of covariance matrices $\Sigma_1, \ldots, \Sigma_N$, under the assumption that it is piecewise constant, *i.e.*, it is often the case that $\Sigma_{i+1} = \Sigma_i$. In order to obtain a convex problem, we use the inverse covariances $X_i = \Sigma_i^{-1}$ as our variables.

The Fused Group Lasso method for this problem involves solving

$$\begin{array}{ll}\text{minimize} & \displaystyle\sum_{i=1}^{N} \mathbf{Tr}(X_i y_i y_i^T) - \log\det X_i + \lambda \sum_{i=1}^{N-1} \|R_i\|_F \\ \text{subject to} & R_i = X_{i+1} - X_i, \quad i = 1, \ldots, N-1,\end{array}$$

where our variables are $R_i \in \mathbf{S}^n$, $i = 1, \ldots, N-1$, and $X_i \in \mathbf{S}_+^n$, $i = 1, \ldots, N$. Here,

$$\|R_i\|_F = \sqrt{\mathbf{Tr}(R_i^T R_i)}$$

is the Frobenius norm of R_i. Let $X_1^\star, \ldots, X_N^\star, R_1^\star, \ldots, R_{N-1}^\star$ denote an optimal point, our estimates of $\Sigma_1, \ldots, \Sigma_N$ are $(X_1^\star)^{-1}, \ldots, (X_N^\star)^{-1}$.

ADMM steps. It is easy to see that steps (9) and (10) simplify for this problem. Step (9) requires solving

$$X_i^{k+1} := \operatorname*{argmin}_{X_i \succ 0}\{\Phi_i(X_i) + (\rho/2)\|X_i - Z_i^k + U_i^k\|_2^2\},$$

where

$$\Phi_i(X_i) = \mathbf{Tr}(X_i y_i y_i^T) - \log\det X_i.$$

This update can be solved analytically, as follows.

(1) Compute the eigenvalue decomposition of

$$\rho\left(Z_i^k - U_i^k\right) - y_i y_i^T = Q \Lambda Q^T$$

where $\Lambda = \mathbf{diag}(\lambda_1, \ldots, \lambda_n)$.

(2) Now let

$$\mu_j := \frac{\lambda_j + \sqrt{\lambda_j^2 + 4\rho}}{2\rho}, \quad j = 1, \ldots, n.$$

(3) Finally, we set

$$X_i^{k+1} = Q\,\mathbf{diag}(\mu_1, \ldots, \mu_n) Q^T.$$

For details of this derivation, see Section 6.5 in Boyd et al. [2011].

Step (10) is

$$R_i^{k+1} := \operatorname*{argmin}_{R_i}\{\lambda\|R_i\|_F + (\rho/2)\|R_i - S_i^k + T_i^k\|_2^2\},$$

which simplifies to

$$R_i^{k+1} = \mathcal{S}_{\lambda/\rho}(S_i^k - T_i^k),$$

where $\mathcal{S}_\kappa$ is a matrix soft threshold operator, defined as

$$\mathcal{S}_\kappa(A) = (1 - \kappa/\|A\|_F)_+ A, \quad \mathcal{S}_\kappa(0) = 0.$$

Variations. As with ℓ_1 mean filtering, we can replace the Frobenius norm penalty with a componentwise vector ℓ_1-norm penalty on R_i to get the problem

$$\begin{array}{ll}\text{minimize} & \displaystyle\sum_{i=1}^{N} \mathbf{Tr}(X_i y_i y_i^T) - \log\det X_i + \lambda \sum_{i=1}^{N-1} \|R_i\|_1 \\ \text{subject to} & R_i = X_{i+1} - X_i, \quad i = 1, \ldots, N-1,\end{array}$$

with variables $R_1, \ldots, R_{N-1} \in \mathbf{S}^n$, and $X_1, \ldots, X_N \in \mathbf{S}_+^n$, and where

$$\|R\|_1 = \sum_{j,k} |R_{jk}|.$$

Again, the ADMM updates are the same, the only difference is that in step (10) we replace matrix soft thresholding with a componentwise soft threshold, *i.e.*,

$$(R_i^{k+1})_{l,m} = \mathcal{S}_{\lambda/\rho}((S_i^k - T_i^k)_{l,m}),$$

for $l = 1, \ldots, n$, $m = 1, \ldots, n$.

4.3 ℓ_1 Mean and variance filtering

Consider a sequence of vector random variables

$$Y_i \sim \mathcal{N}(\bar{y}_i, \Sigma_i), \quad i = 1, \ldots, N,$$

where $\bar{y}_i \in \mathbf{R}^n$ is the mean, and $\Sigma_i \in \mathbf{S}_+^n$ is the covariance matrix for Y_i. We assume that the mean and covariance matrix of the process is unknown. Given observations $y_1, \ldots, y_N$, our goal is to estimate the mean and the sequence of covariance matrices $\Sigma_1, \ldots, \Sigma_N$, under the assumption that they are piecewise constant, *i.e.*, it is often the case that $\bar{y}_{i+1} = \bar{y}_i$ and $\Sigma_{i+1} = \Sigma_i$. To obtain a convex optimization problem, we use

$$X_i = -\frac{1}{2}\Sigma_t^{-1}, \quad m_i = \Sigma_t^{-1} x_i,$$

as our variables. In the Fused Group Lasso method, we obtain our estimates by solving

$$\begin{array}{ll}\text{minimize} & \displaystyle\sum_{i=1}^{N} -(1/2)\log\det(-X_i) - \mathbf{Tr}(X_i y_i y_i^T) \\ & \quad -m_i^T y_i - (1/4)\,\mathbf{Tr}(X_i^{-1} m_i m_i^T) \\ & \quad +\lambda_1 \displaystyle\sum_{i=1}^{N-1} \|r_i\|_2 + \lambda_2 \sum_{i=1}^{N-1} \|R_i\|_F \\ \text{subject to} & r_i = m_{i+1} - m_i, \quad i = 1, \ldots, N-1, \\ & R_i = X_{i+1} - X_i, \quad i = 1, \ldots, N-1,\end{array}$$

with variables $r_1, \ldots, r_{N-1} \in \mathbf{R}^n$, $m_1, \ldots, m_N \in \mathbf{R}^n$, $R_1, \ldots, R_{N-1} \in \mathbf{S}^n$, and $X_1, \ldots, X_N \in \mathbf{S}^n_+$.

ADMM steps. This problem is also in the form (6), however, as far as we are aware, there is no analytical formula for steps (9) and (10). To carry out these updates, we must solve semidefinite programs (SDPs), for which there are a number of efficient and reliable software packages (Toh et al. [1999], Sturm [1999]).

5. NUMERICAL EXAMPLE

In this section we solve an instance of ℓ_1 mean filtering with $n = 1$, $\Sigma = 1$, and $N = 400$, using the Fused Group Lasso method. To improve convergence of the ADMM algorithm, we use over-relaxation with $\alpha = 1.8$, see Boyd et al. [2011]. The parameter λ is chosen as approximately 10% of $\lambda_{\max}$, where $\lambda_{\max}$ is the largest value that results in a non-constant mean estimate. Here, $\lambda_{\max} \approx 108$ and so $\lambda = 10$. We use an absolute plus relative error stopping criterion, with $\epsilon^{\text{abs}} = 10^{-4}$ and $\epsilon^{\text{rel}} = 10^{-3}$. Figure 1 shows convergence of the primal and dual residuals. The resulting estimates of the means are shown in Figure 2.

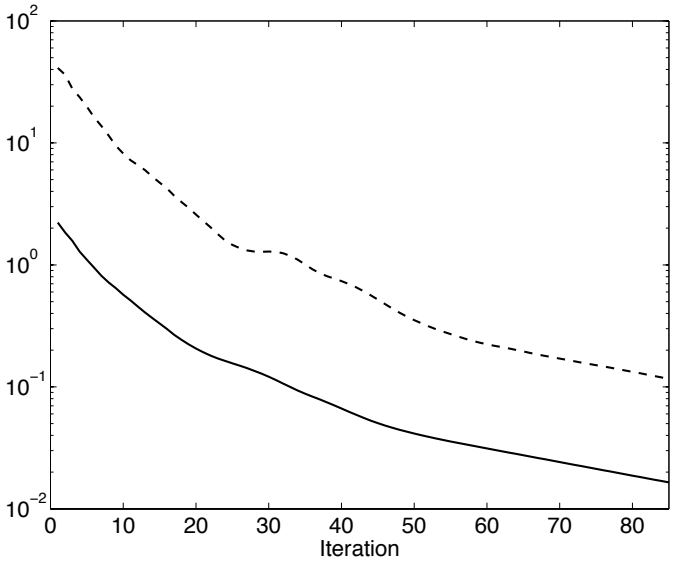

Fig. 1. Residual convergence: Primal residual e_p (solid line), and dual residual e_d (dashed line).

We solved the same ℓ_1 mean filtering problem using CVX, a package for specifying and solving convex optimization problems (Grant and Boyd [2011]). CVX calls generic SDP solvers SeDuMi (Toh et al. [1999]) or SDPT3 (Sturm [1999]) to solve the problem. While these solvers are reliable for wide classes of optimization problems, and exploit sparsity in the problem formulation, they are not customized for particular problem families, such as ours. The computation time for CVX is approximately 20 seconds. Our ADMM algorithm (implemented in C), took 2.2 *milliseconds* to produce the same estimates. Thus, our algorithm is approximately 10000 times faster compared with generic optimization packages. Indeed, our implementation does *not* exploit the fact that steps 1 and 3 of ADMM can be implemented independently in parallel for each measurement. Parallelizing steps 1 and 3 of the computation can lead to further speedups. For example, simple multi-threading on a quad-core CPU would result in a further 4× speed-up.

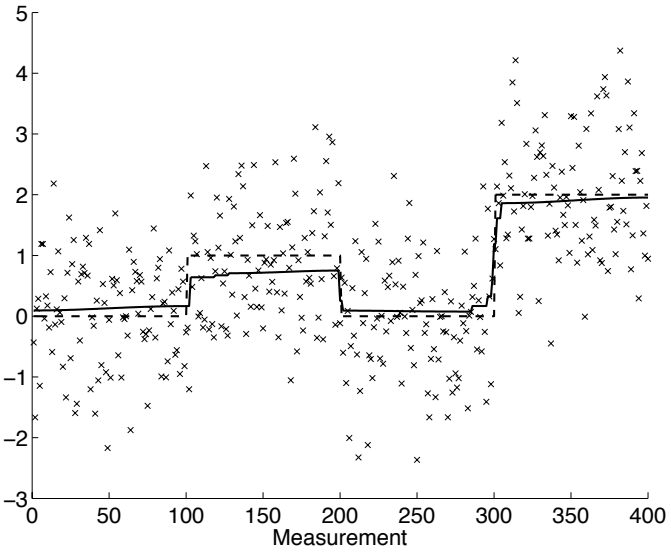

Fig. 2. Estimated means (solid line), true means (dashed line) and measurements (crosses).

6. CONCLUSIONS

In this paper we derived an efficient and scalable method for an optimization problem (6) that has a variety of applications in control and estimation. Our custom method exploits the structure of the problem via a distributed optimization framework. In many applications, each step of the method is a simple update that typically involves solving a set of linear equations, matrix multiplication, or thresholding, for which there are exceedingly efficient libraries. In numerical examples we have shown that we can solve problems such as ℓ_1 mean and variance filtering many orders of magnitude faster than generic optimization solvers such as SeDuMi or SDPT3.

The only tuning parameter for our method is the regularization parameter ρ. Finding an optimal ρ is not a straightforward problem, but Boyd et al. [2011] contains many heuristics that work well in practice. For the ℓ_1 mean filtering example, we find that setting $\rho \approx \lambda$ works well, but we do not have a formal justification.

REFERENCES

O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation

for multivariate gaussian or binary data. *Journal of Machine Learning Research*, 9:485–516, 2008.

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.

P. L. Combettes and J. C. Pesquet. A Douglas-Rachford splitting approach to nonsmooth convex variational signal recovery. *Selected Topics in Signal Processing, IEEE Journal of*, 1(4):564 –574, dec. 2007. ISSN 1932-4553. doi: 10.1109/JSTSP.2007.910264.

A. P. Dempster. Covariance selection. *Biometrics*, 28(1): 157–175, 1972.

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

J. Friedman, T. Hastie, and R. Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming, version 1.21. `http://cvxr.com/cvx`, April 2011.

S. J. Kim, K. Koh, S. Boyd, and D. Gorinevsky. l_1 trend filtering. *SIAM Review*, 51(2):339–360, 2009.

H. Ohlsson, L. Ljung, and S. Boyd. Segmentation of arx-models using sum-of-norms regularization. *Automatica*, 46:1107 – 1111, April 2010.

L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60:259–268, November 1992. ISSN 0167-2789. doi: http://dx.doi.org/10.1016/0167-2789(92)90242-F.

Simon Setzer. Operator splittings, bregman methods and frame shrinkage in image processing. *International Journal of Computer Vision*, 92(3):265–280, 2011.

J. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11:625–653, 1999. Software available at `http://sedumi.ie.lehigh.edu/`.

R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67 (Part 1):91–108, 2005.

K. Toh, M. Todd, and R. Tütüncü. SDPT3—A Matlab software package for semidefinite programming, version 1.3. *Optimization Methods and Software*, 11(1):545–581, 1999.

B. Wahlberg, C. R. Rojas, and M. Annergren. On l_1 mean and variance filtering. *Proceedings of the Forty-Fifth Asilomar Conference on Signals, Systems and Computers*, 2011. arXiv/1111.5948.