Ann Oper Res (2010) 174: 3–17
DOI 10.1007/s10479-008-0348-8

Image denoising via solution paths

Li Wang · Ji Zhu

Published online: 1 May 2008

Abstract Many image denoising methods can be characterized as minimizing "loss + penalty," where the "loss" measures the fidelity of the denoised image to the data, and the "penalty" measures the smoothness of the denoising function. In this paper, we propose two models that use the L_1-norm of the pixel updates as the penalty. The L_1-norm penalty has the advantage of changing only the noisy pixels, while leaving the non-noisy pixels untouched. We derive efficient algorithms that compute entire solution paths of these L_1-norm penalized models, which facilitate the selection of a balance between the "loss" and the "penalty."

Keywords Image denoising · L_1-norm penalty · PCA · Regularization · Solution paths

1 Introduction

1.1 Background

Image denoising problems arise in many engineering fields and applied physics, because in practice, images can often be contaminated when they are acquired by sensors or transferred in communication channels. This problem has been studied for decades, and researchers have proposed several categories of methods to tackle this problem, such as filtering methods (Harwood et al. 1987; Schulze and Pearce 1994; Weeks 1996), wavelet-based methods (Bruce and Gao 1996), principal component analysis (PCA)-based methods (Mika et al. 1999; Takahashi and Kurita 2002) and sparse coding (SC) shrinkage methods (Hyvarinen et al. 1998; Shang et al. 2006). These methods apply in different contexts and use different strategies for denoising the contaminated image. When prior knowledge of the distribution of an image is not available, filtering methods are often used. Filtering methods tend to blur an image, however, and edges in the image are not well preserved. Wavelet-based methods are proposed to overcome this problem: they decompose a contaminated image using

L. Wang · J. Zhu (✉)
Department of Statistics, University of Michigan, 439 West Hall, 1085 South University Ave, Ann Arbor, MI 48109, USA
e-mail: jizhu@umich.edu

Fig. 1 The original face image (*left panel*) and the corresponding image corrupted by impulse noises (*right panel*)

wavelet bases and reconstruct the image by shrinking some of the wavelet coefficients; in this fashion, edges are often better preserved. PCA-based methods and SC shrinkage methods are used when some prior knowledge of the statistical properties of the image are available. These statistical properties are often obtained from a collection of images, which share common characteristics of the target image. The methods we propose belong to the category of PCA-based methods.

In most of the current literature, images are often assumed to be contaminated by an additive Gaussian noise. Additive Gaussian noise, which comes from the background, is often observed in analog systems. In that setting, *all* pixels are corrupted by normally distributed noise, and the magnitude of the noise is small. In this paper, we consider a different type of noise, the "impulse noise". In the impulse noise setting, only a portion of the pixels are (completely) corrupted while the values of other pixels remain accurate. Impulse noise is commonly found in digital systems, where digital images are corrupted because of, for example, malfunctioning pixels in the camera sensors, faulty memory locations in the hardware, transmission errors, analog-to-digital conversion errors, etc. Impulse noise poses new challenges to traditional image denoising methods, because the noise is no longer Gaussian; it is "sparse" and "spiky." Figure 1 shows an example of an original face image and a version corrupted by "impulse noise." Ideally, good denoising methods for impulse noise should be able to identify the corrupted pixels and correct only these pixels.

1.2 PCA-based methods

Images can be represented by vectors. From a collection of images which share common characteristics, we can calculate their principal components (PC), which contain important information of the images. PCA-based methods project the corrupted image to the space consisting of these PC vectors.

Let $\boldsymbol{x} \in \mathbb{R}^n$ denote the corrupted image and $\boldsymbol{U} \in \mathbb{R}^{n\times p}$ be the matrix with PCs as its columns. Without loss of generality, we assume both $\boldsymbol{x}$ and the column vectors of $\boldsymbol{U}$ are centered and standardized. Classical PCA-based methods project $\boldsymbol{x}$ to the column space of $\boldsymbol{U}$ via the least squares regression, i.e.,

$$\min_{\boldsymbol{\beta}} \|\boldsymbol{x} - \boldsymbol{U}\boldsymbol{\beta}\|_2^2, \tag{1}$$

and the denoised image is constructed as $\boldsymbol{U}\boldsymbol{\beta}$. The least squares estimates often have low bias but high variance, which hurts the accuracy of the denoised image. In this paper, we propose regularization models that have the following form:

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \ell(\boldsymbol{x} + \boldsymbol{\alpha}, \boldsymbol{U}\boldsymbol{\beta}) + \lambda(\|\boldsymbol{\alpha}\|_1 + \|\boldsymbol{\beta}\|_1), \tag{2}$$

where $\ell(\cdot,\cdot)$ is a loss function describing the discrepancy between $\boldsymbol{x}+\boldsymbol{\alpha}$ and $\boldsymbol{U}\boldsymbol{\beta}$, $\|\boldsymbol{\alpha}\|_1$ and $\|\boldsymbol{\beta}\|_1$ are L_1-norm penalties of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, equaling $|\alpha_1|+\cdots+|\alpha_n|$ and $|\beta_1|+\cdots+|\beta_p|$, and λ is a tuning parameter controlling the balance between the loss and the penalty.

There are two major differences between our models and previous models:

- The introduction of $\boldsymbol{\alpha}$: $\boldsymbol{\alpha}$ is a n-dimensional vector, and it is interpreted as a pixel-by-pixel updating vector for the "impulse noise" contaminated image. Hence our methods deliver two denoised images, $\boldsymbol{x}+\boldsymbol{\alpha}$ and $\boldsymbol{U}\boldsymbol{\beta}$, while previous models only deliver one denoised image $\boldsymbol{U}\boldsymbol{\beta}$, which is a linear combination of the PC images. As we will see later, $\boldsymbol{x}+\boldsymbol{\alpha}$ is often a better denoised image than $\boldsymbol{U}\boldsymbol{\beta}$ when the noise is the impulse type.
- The usage of the L_1-norm penalty: The L_1-norm (LASSO) penalty sets some of the elements in $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ exactly equal to zero, i.e., sparse property (Mallat and Zhang 1993; Tibshirani 1996; Chen et al. 1998). When constructing the denoised image, some of the PC images are important, and some of the PC images may not be important. The L_1-norm of $\boldsymbol{\beta}$ helps eliminate unimportant PC images. More importantly, since we consider the impulse type noises, i.e., some pixels of the original image are corrupted but other pixels are good, the L_1-norm of $\boldsymbol{\alpha}$ helps identify and repair only the corrupted pixels and leave the uncorrupted ones untouched. This is related to the variable selection problem in statistics, which has been studied extensively in the literature, for example, see George and McCulloch (1993), Breiman (1995), Tibshirani (1996), George and Foster (2000), and Fan and Li (2001).

As we will see in numerical studies, these new models show promising results in denoising the impulse type noise.

The rest of the paper is organized as follows: In Sect. 2, we propose two models for image denoising that use the L_1-norm of the pixel updates as the penalty. In Sect. 3, we derive efficient algorithms that compute the entire solution paths of these two models. In Sect. 4, we present numerical results on a real image dataset. We conclude the paper with Sect. 5.

2 Models

Since the impulse type noise tends to have a heavy-tail distribution, and the squared error loss is not robust to outliers, we consider the least absolute deviation (LAD) loss, in addition to the least squares (LS) loss. Specifically, we consider the following regularization models for denoising impulsely contaminated images:

$$\text{LAD-LASSO:}\quad \min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \|\boldsymbol{x}+\boldsymbol{\alpha}-\boldsymbol{U}\boldsymbol{\beta}\|_1+\lambda(\|\boldsymbol{\alpha}\|_1+\|\boldsymbol{\beta}\|_1), \tag{3}$$

$$\text{LS-LASSO:}\quad \min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \|\boldsymbol{x}+\boldsymbol{\alpha}-\boldsymbol{U}\boldsymbol{\beta}\|_2^2+\lambda(\|\boldsymbol{\alpha}\|_1+\|\boldsymbol{\beta}\|_1). \tag{4}$$

To further improve models (3) and (4), we apply the adaptive idea which has been used in Breiman (1995), Shen and Ye (2002), Zhao and Yu (2006) and Zou (2006), i.e., to penalize different components in $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ differently:

$$\text{LAD-Adaptive-LASSO:}\quad \min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \|\boldsymbol{x}+\boldsymbol{\alpha}-\boldsymbol{U}\boldsymbol{\beta}\|_1+\lambda\left(\sum_{i=1}^{n}\frac{|\alpha_i|}{|\alpha_i^0|}+\sum_{j=1}^{p}\frac{|\beta_j|}{|\beta_j^0|}\right), \tag{5}$$

$$\text{LS-Adaptive-LASSO:} \quad \min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \|\boldsymbol{x}+\boldsymbol{\alpha}-\boldsymbol{U}\boldsymbol{\beta}\|_2^2+\lambda\left(\sum_{i=1}^{n}\frac{|\alpha_i|}{|\alpha_i^0|}+\sum_{j=1}^{p}\frac{|\beta_j|}{|\beta_j^0|}\right), \tag{6}$$

where $|\alpha_i^0|$ and $|\beta_j^0|$ can be regarded as pre-fixed adaptive weights. The intuition is that for unimportant PC images and uncorrupted pixels, we would like the corresponding weights $|\beta_j^0|$ and $|\alpha_i^0|$ to be small, hence β_j and α_i are heavily penalized, while for important PC images and corrupted pixels, we would like the corresponding weights $|\beta_j^0|$ and $|\alpha_i^0|$ to be big, hence β_j and α_i are lightly penalized. How to pre-specify the weights $|\beta_j^0|$ and $|\alpha_i^0|$ from the data is discussed in Sect. 4.

As in every regularization problem, choice of the regularization parameter λ is critical. In practice, people usually pre-specify a finite set of candidate values for λ that cover a wide range, then either use a separate validation dataset or certain model selection criterion to pick a value for λ that gives the best performance among the pre-specified set. For detailed description of different model selection criteria for image denoising, we refer the readers to Thompson et al. (1991). There is no theoretical guidance on how to pick the candidate set of the regularization parameter; people have to rely on either their previous experience or the trial-and-error approach. Since the denoising performance is often sensitive to the value of the regularization parameter, parameter tuning can be very time-consuming and the best parameter can be easily missed.

In the next section, we derive very efficient algorithms that compute the *exact entire solution paths* of $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ as functions of λ, which facilitate selection of the regularization parameter.

3 Solution path algorithms

In this section, we derive efficient algorithms that compute the exact solution paths for LAD-Adaptive-LASSO (5) and LS-Adaptive-LASSO (6). Since the algorithm for LS-Adaptive-LASSO is a simple modification of the LAR/LASSO algorithm (Efron et al. 2004), we focus only on the algorithm for LAD-Adaptive-LASSO. We also note that the naive methods (3) and (4) are special cases of the adaptive methods, with $|\alpha_i^0|=1$ and $|\beta_j^0|=1$.

In the following sections, we use $\boldsymbol{u}_i$ to denote the ith row of the eigen-image matrix $\boldsymbol{U}$.

3.1 Problem setup

Criterion (5) can be re-written in an equivalent way:

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad \sum_{i=1}^{n}\epsilon_i \tag{7}$$

$$\text{subject to} \quad -\epsilon_i \le x_i+\alpha_i-\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{\beta} \le \epsilon_i, \tag{8}$$

$$\epsilon_i \ge 0, \quad i=1,\ldots,n, \tag{9}$$

$$\sum_{i=1}^{n}\frac{|\alpha_i|}{|\alpha_i^0|}+\sum_{j=1}^{p}\frac{|\beta_j|}{|\beta_j^0|} \le s. \tag{10}$$

Notice the absolute value loss is replaced with two linear constraints in (8). Also notice that the tuning parameter λ is replaced by another equivalent tuning parameter s. We are going

to show that the solutions $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are piecewise linear functions with respect to s, which allow us to develop an efficient algorithm to compute the entire solution path. Equations (7)–(10) give the Lagrangian primal function:

$$\begin{aligned} L_p: \quad & \sum_{i=1}^{n} \epsilon_i + \lambda^* \left(\sum_{i=1}^{n} \frac{|\alpha_i|}{|\alpha_i^0|} + \sum_{j=1}^{p} \frac{|\beta_j|}{|\beta_j^0|} - s \right) \\ & + \sum_{i=1}^{n} \eta_i^+ (x_i + \alpha_i - \boldsymbol{u}_i^\top \boldsymbol{\beta} - \epsilon_i) \\ & - \sum_{i=1}^{n} \eta_i^- (x_i + \alpha_i - \boldsymbol{u}_i^\top \boldsymbol{\beta} + \epsilon_i) \\ & - \sum_{i=1}^{n} \gamma_i \epsilon_i, \end{aligned}$$

where $\lambda^*, \eta_i^+, \eta_i^-$ and γ_i are non-negative Lagrangian multipliers. Setting the derivatives of L_p to zero, we arrive at:

$$\frac{\partial}{\partial \boldsymbol{\beta}}: \quad -\sum_{i=1}^{n} (\eta_i^+ - \eta_i^-) u_{ij} + \lambda^* \frac{\operatorname{sign}(\beta_j)}{|\beta_j^0|} = 0, \quad \text{for } j \text{ with } \beta_j \neq 0, \tag{11}$$

$$\frac{\partial}{\partial \boldsymbol{\alpha}}: \quad (\eta_i^+ - \eta_i^-) + \lambda^* \frac{\operatorname{sign}(\alpha_i)}{|\alpha_i^0|} = 0, \quad \text{for } i \text{ with } \alpha_i \neq 0, \tag{12}$$

$$\frac{\partial}{\partial \epsilon_i}: \quad 1 - \eta_i^+ - \eta_i^- - \gamma_i = 0, \quad i = 1, \ldots, n, \tag{13}$$

and the complementary slackness conditions are:

$$\eta_i^+ (x_i + \alpha_i - \boldsymbol{u}_i^\top \boldsymbol{\beta} - \epsilon_i) = 0, \quad i = 1, \ldots, n, \tag{14}$$

$$\eta_i^- (x_i + \alpha_i - \boldsymbol{u}_i^\top \boldsymbol{\beta} + \epsilon_i) = 0, \quad i = 1, \ldots, n, \tag{15}$$

$$\gamma_i \epsilon_i = 0, \quad i = 1, \ldots, n. \tag{16}$$

Since the Lagrange multipliers must be non-negative, we see from (13) that $0 \leq \eta_i^+, \eta_i^- \leq 1$. We can also see that (8) and (13)–(16) lead to the following:

$$\begin{array}{lllllll} x_i + \alpha_i - \boldsymbol{u}_i^\top \boldsymbol{\beta} > 0 & \Rightarrow & \epsilon_i > 0, & \gamma_i = 0, & \eta_i^+ = 1, & \eta_i^- = 0; \\ x_i + \alpha_i - \boldsymbol{u}_i^\top \boldsymbol{\beta} < 0 & \Rightarrow & \epsilon_i > 0, & \gamma_i = 0, & \eta_i^+ = 0, & \eta_i^- = 1; \\ x_i + \alpha_i - \boldsymbol{u}_i^\top \boldsymbol{\beta} = 0 & \Rightarrow & \epsilon_i = 0, & \gamma_i \in [0,1], & \eta_i^+ \in [0,1], & \eta_i^- \in [0,1]. \end{array}$$

We define $\eta_i = \eta_i^+ - \eta_i^-$. Hence, using these relationships, we can define the following five sets that will be used later when we calculate the solution path of LAD-Adaptive-LASSO:

- $\mathcal{E} = \{i : x_i + \alpha_i - \boldsymbol{u}_i^\top \boldsymbol{\beta} = 0, -1 \leq \eta_i \leq 1\}$ (Elbow)
- $\mathcal{R} = \{i : x_i + \alpha_i - \boldsymbol{u}_i^\top \boldsymbol{\beta} > 0, \eta_i = 1\}$ (Right of the elbow)
- $\mathcal{L} = \{i : x_i + \alpha_i - \boldsymbol{u}_i^\top \boldsymbol{\beta} < 0, \eta_i = -1\}$ (Left of the elbow)
- $\mathcal{V} = \{j : \beta_j \neq 0\}$ (Active set for $\boldsymbol{\beta}$)
- $\mathcal{W} = \{i : \alpha_i \neq 0\}$ (Active set for $\boldsymbol{\alpha}$)

For pixels in $\mathcal{R}$ and $\mathcal{L}$, their η_i are determined. Therefore, we will focus on pixels in $\mathcal{E}$.

The basic idea of our algorithm is as follows: We start with $s = 0$ and increase it, keeping track of the location of all pixels relative to the elbow and also of the magnitude of the fitted coefficients β_j and α_i along the way. As s increases, by continuity, points in the elbow $\mathcal{E}$ must linger on it. Since all points at the elbow have $x_i + \alpha_i - \boldsymbol{u}_i^{\mathsf{T}}\boldsymbol{\beta} = 0$, we can establish a path for $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$. The elbow set will stay stable until either a new pixel comes to the elbow or a non-zero fitted coefficient has dropped to zero.

3.1.1 Initialization

Initially, when $s = 0$, we can see from (8)–(10) that $\epsilon_i = |x_i|$, hence the initial dual variables η_i's and the initial sets $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{L}$ can be determined.

When $s \to 0^+$, one of the α_i's or β_j's will become non-zero. Let $\lambda_\beta^* = \max_j |\beta_j^0 \sum_{i=1}^n \eta_j u_{ij}|$ and $\lambda_\alpha^* = \max_i |\alpha_i^0 \eta_i|$, then from (11)–(12), we can see

- If $\lambda_\beta^* > \lambda_\alpha^*$, the initial $\mathcal{V} = \{j^* = \arg\max_j |\beta_j^0 \sum_{i=1}^n \eta_j u_{ij}|\}$, $\mathcal{W} = \emptyset$, $\operatorname{sign}(\beta_{j^*}) = \operatorname{sign}(\sum_{i=1}^n \eta_{j^*} u_{ij^*})$, and $\lambda^* = \lambda_\beta^*$.
- If $\lambda_\beta^* < \lambda_\alpha^*$, the initial $\mathcal{V} = \emptyset$, $\mathcal{W} = \{i^* = \arg\max_i |\alpha_i^0 \eta_i|\}$, $\operatorname{sign}(\alpha_{i^*}) = -\operatorname{sign}(\eta_{i^*})$, and $\lambda^* = \lambda_\alpha^*$.

In the following, we use ℓ to indicate the step number, with the initial $\ell = 0$.

3.1.2 The solution paths

When s is small, the constraint (10) is active, i.e., $\sum_{i=1}^n |\alpha_i|/|\alpha_i^0| + \sum_{j=1}^p |\beta_j|/|\beta_j^0| = s$. When s increases to a certain value, say s^*, this constraint will become inactive, and the solution will not change beyond the value of s^*. This corresponds to $\lambda = 0$ in (5). Suppose for a value $s < s^*$, we have the solution $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, hence $\mathcal{E}$, $\mathcal{R}$, $\mathcal{L}$, $\mathcal{V}$, and $\mathcal{W}$ are also known. Then $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ have to satisfy the following equations:

$$
\begin{aligned}
x_i + \alpha_i - \sum_{j\in\mathcal{V}} u_{ij}\beta_j = 0, &\quad \text{for } i \in \mathcal{E} \text{ and } i \in \mathcal{W},\\
x_i - \sum_{j\in\mathcal{V}} u_{ij}\beta_j = 0, &\quad \text{for } i \in \mathcal{E} \text{ and } i \notin \mathcal{W},\\
\sum_{i\in\mathcal{W}} \frac{|\alpha_i|}{|\alpha_i^0|} + \sum_{j\in\mathcal{V}} \frac{|\beta_j|}{|\beta_j^0|} = s.&
\end{aligned}
$$

This linear system consists of $|\mathcal{E}| + 1$ equations and $|\mathcal{V}| + |\mathcal{W}|$ unknowns. We also notice that the linear system (11)–(12) consists of $|\mathcal{V}| + |\mathcal{W}|$ equations and $|\mathcal{E}| + 1$ unknowns. Therefore, to satisfy both systems, we must have $|\mathcal{V}| + |\mathcal{W}| = |\mathcal{E}| + 1$.

When s increases by a small enough amount, the sets $\mathcal{E}$, $\mathcal{R}$, $\mathcal{L}$, $\mathcal{V}$ and $\mathcal{W}$ will not change due to their continuity with respect to s, and the structure of the above linear system will not change. Taking right derivatives with respect to s, we have

$$
\frac{\Delta\alpha_i}{\Delta s} - \sum_{j\in\mathcal{V}} u_{ij}\frac{\Delta\beta_j}{\Delta s} = 0, \quad \text{for } i \in \mathcal{E} \text{ and } i \in \mathcal{W}, \tag{17}
$$

$$
\sum_{j\in\mathcal{V}} u_{ij}\frac{\Delta\beta_j}{\Delta s} = 0, \quad \text{for } i \in \mathcal{E} \text{ and } i \notin \mathcal{W}, \tag{18}
$$

$$\sum_{i\in\mathcal{W}} \frac{\operatorname{sign}(\alpha_i^\ell)}{|\alpha_i^0|}\frac{\Delta\alpha_i}{\Delta s} + \sum_{j\in\mathcal{V}} \frac{\operatorname{sign}(\beta_j^\ell)}{|\beta_j^0|}\frac{\Delta\beta_j}{\Delta s} = 1. \tag{19}$$

Since $|\mathcal{E}| + 1 = |\mathcal{V}| + |\mathcal{W}|$, $\Delta\alpha_i/\Delta s$ and $\Delta\beta_j/\Delta s$ are constant and can be uniquely determined, assuming the linear system is non-singular. Therefore, if s increases by a small enough amount, α_i and β_j change linearly in s

$$\alpha_i = \alpha_i^\ell + (s - s^\ell)\frac{\Delta\alpha_i}{\Delta s}, \quad \text{for } i \in \mathcal{W}, \tag{20}$$

$$\beta_j = \beta_j^\ell + (s - s^\ell)\frac{\Delta\beta_j}{\Delta s}, \quad \text{for } j \in \mathcal{V}, \tag{21}$$

and the residual $r_i = x_i + \alpha_i - \boldsymbol{u}_i^\mathrm{T}\boldsymbol{\beta}$ changes as

$$r_i = r_i^\ell + (\alpha_i - \alpha_i^\ell) - \sum_{j\in\mathcal{V}} (\beta_j - \beta_j^\ell)u_{ij}. \tag{22}$$

When the increase in s is big enough, one of the $\mathcal{E}$, $\mathcal{V}$ and $\mathcal{W}$ sets will change, and the structure of the linear system will also change.

To identify changes in the structure of the linear system, we define three types of *events*, corresponding to the changes in $\mathcal{E}$, $\mathcal{V}$ and $\mathcal{W}$.

- A pixel hits the elbow $\mathcal{E}$ from $\mathcal{R}$ or $\mathcal{L}$, i.e., a residual $x_i + \alpha_i - \boldsymbol{u}_i^\mathrm{T}\boldsymbol{\beta}$ changes from non-zero to zero.
- A coefficient α_i changes from non-zero to zero.
- A coefficient β_j changes from non-zero to zero.

Given s^ℓ, (20)–(22) allow us to compute $s^{\ell+1}$, the value of s at which the next *event* will occur. This will be the smallest s larger than s^ℓ, such that either ϵ_i for $i \notin \mathcal{E}$ reaches zero, or one of the coefficients α_i for $i \in \mathcal{W}$ or β_j for $j \in \mathcal{V}$ reaches zero.

When an *event* occurs, by the definition of an *event*, the condition $|\mathcal{E}| + 1 = |\mathcal{V}| + |\mathcal{W}|$ no longer holds. Therefore, to make the KKT conditions satisfied, we need to take some *action*. We define three types of *actions*:

- remove a pixel from $\mathcal{E}$;
- add a new coefficient into $\mathcal{V}$;
- add a new coefficient into $\mathcal{W}$.

The choice can be determined by solving the dual variables using (11)–(12).

Let $\mathcal{E}^*$, $\mathcal{R}^*$, $\mathcal{L}^*$, $\mathcal{V}^*$ and $\mathcal{W}^*$ denote the sets immediately after the $(\ell+1)$th *event* has occurred. Notice that $|\mathcal{E}^*| = |\mathcal{V}^*| + |\mathcal{W}^*|$. For pixels in $\mathcal{R}^*$ and $\mathcal{L}^*$, the values of η_i are known and fixed, so we have

$$\sum_{i\in\mathcal{E}^*} (\eta_i - \eta_i^\ell)u_{ij} = (\lambda^* - \lambda^{*\ell})\frac{\operatorname{sign}(\beta_j^\ell)}{|\beta_j^0|}, \quad \forall j \in \mathcal{V}^*,$$

$$-(\eta_i - \eta_i^\ell) = (\lambda^* - \lambda^{*\ell})\frac{\operatorname{sign}(\alpha_i^\ell)}{|\alpha_i^0|}, \quad \forall i \in \mathcal{W}^*.$$

To simplify, let $\frac{\Delta\eta_i}{\Delta\lambda^*} = (\eta_i - \eta_i^\ell)/(\lambda^* - \lambda^{*\ell})$. Then

$$\sum_{i\in\mathcal{E}^*} \frac{\Delta\eta_i}{\Delta\lambda^*}u_{ij} = \frac{\operatorname{sign}(\beta_j^\ell)}{|\beta_j^0|}, \quad \forall j \in \mathcal{V}^*,$$

$$-\frac{\Delta\eta_i}{\Delta\lambda^*} = \frac{\text{sign}(\alpha_i)}{|\alpha_i^0|}, \quad \forall i \in \mathcal{W}^*.$$

There are $|\mathcal{V}^*|+|\mathcal{W}^*|$ equations and $|\mathcal{E}^*|$ unknowns. Since $|\mathcal{E}^*| = |\mathcal{V}^*|+|\mathcal{W}^*|$, we can solve for $\Delta\eta_i/\Delta\lambda^*$, and we have

$$\eta_i = \eta_i^\ell + (\lambda^* - \lambda^{*\ell})\frac{\Delta\eta_i}{\Delta\lambda^*}, \quad \text{for } i \in \mathcal{E}^*. \tag{23}$$

Thus for $\lambda^{*\ell} > \lambda^* > \lambda^{*(\ell+1)}$, the η_i's proceed linearly in λ^*.

Regarding adding a coefficient into $\mathcal{V}$ or $\mathcal{W}$, from (11) and (12), we can see that when λ^* decreases, it corresponds to

$$\sum_{i\in\mathcal{E}^*}\left(\eta_i^\ell + (\lambda^* - \lambda^{*\ell})\frac{\Delta\eta_i}{\Delta\lambda^*}\right)u_{ij} + \sum_{i\in\mathcal{R}^*} u_{ij} - \sum_{i\in\mathcal{L}^*} u_{ij}$$
$$= \lambda^*\frac{\text{sign}(\beta_j)}{|\beta_j^0|}, \quad \text{for some } j \notin \mathcal{V}^*, \tag{24}$$

or

$$-\left(\eta_i^\ell + (\lambda^* - \lambda^{*\ell})\frac{\Delta\eta_i}{\Delta\lambda^*}\right) = \lambda^*\frac{\text{sign}(\alpha_i)}{|\alpha_i^0|}, \quad \text{for some } i \notin \mathcal{W}^*. \tag{25}$$

Equations (23)–(25) allow us to compute $\lambda^{*(\ell+1)}$, the largest λ^* smaller than $\lambda^{*\ell}$, such that either one of the η_i for $i \in \mathcal{E}$ reaches 1 or -1, or one of the coefficients α_i for $i \notin \mathcal{W}$ or β_j for $j \notin \mathcal{V}$ joins the active set.

Once an *action* is taken, we restore $|\mathcal{E}|+1 = |\mathcal{V}|+|\mathcal{W}|$. The algorithm keeps increasing s and alternates between reaching the next *event* and taking an *action*, until λ^* reduces to 0.

3.1.3 The computational cost

The major computational cost for updating the solutions at any step ℓ involves two things: solving the primal system (17)–(19) and solving the dual system (23)–(25). Since there are $|\mathcal{E}|+1$ unknowns in each system, and $|\mathcal{E}|$ can be as large as n, it seems that the computational cost is expensive. However, by taking advantage of the special structure in these two systems, we can significantly reduce the computational cost. From (17), we can write

$$\frac{\Delta\alpha_i}{\Delta s} = \sum_{j\in\mathcal{V}} u_{ij}\frac{\Delta\beta_j}{\Delta s}, \quad \text{for } i \in \mathcal{W}.$$

Plug it into (19), we then get a system with only $|\mathcal{V}|$ unknowns. Similarly, we can simplify the dual system to be the same size. Since the sets differ by only one element between consecutive *events*, using inverse updating and downdating, the computational complexity is only $O(|\mathcal{V}|^2)$. It is hard to predict the number of steps in the algorithm, but according to our experience, the total number of steps taken by the algorithm is on average $O(n)$. Since $|\mathcal{V}|$ is upper bounded by p, the overall computational cost is $O(np^2)$.

4 Real data

In this section, we consider an application to a real world dataset. The dataset came from the Max-Planck Institute (MPI) face database (Troje and Bulthoff 1996), which contains face

Table 1 The results are averages over the 20 testing images. The numbers in the parentheses are the corresponding standard errors. "PC-Projection" is for the denoised image $\boldsymbol{U}\boldsymbol{\beta}$, and "pixel-by-pixel" is for the denoised image $\boldsymbol{x}+\boldsymbol{\alpha}$. "LAD" is for the LAD-LASSO method (3), and "LS" is for the LS-LASSO method (4). "LAD-A" and "LS-A" are the corresponding adaptive version (5) and (6). "OLS" is the ordinary least squares method, and "Ideal" is the "idealistic" method

	PC-projection				Pixel-by-pixel				OLS	Ideal
	LS	LAD	LS-A	LAD-A	LS	LAD	LS-A	LAD-A		
Scenario I: "complete corruption"										
MSE	222.9	234.0	222.4	227.5	187.2	188.1	135.3	115.1	2085.3	203.2
	(13.9)	(14.8)	(12.7)	(14.4)	(12.1)	(12.2)	(9.8)	(12.9)	(18.2)	(12.2)
Scenario II: "incomplete corruption"										
MSE	221.5	225.1	216.9	218.3	163.8	161.5	149.2	147.3	624.9	203.3
	(14.2)	(14.2)	(13.1)	(13.5)	(10.7)	(12.3)	(9.3)	(11.7)	(14.5)	(12.2)
Scenario III: "block corruption"										
MSE	230.4	233.8	224.5	231.5	153.3	152.3	140.6	140.1	284.7	210.0
	(14.6)	(14.6)	(13.4)	(15.0)	(9.8)	(12.0)	(8.6)	(12.3)	(18.2)	(12.7)

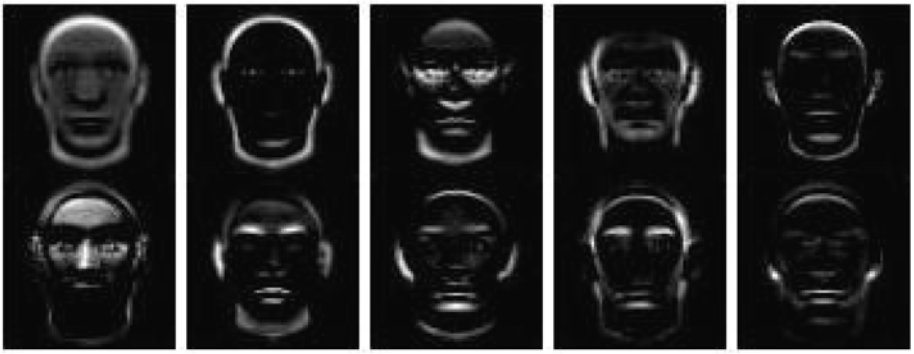

Fig. 2 The first 10 PC-images

Fig. 3 "Complete corruption" scenario. The denoised images using the OLS method (*left panel*) and the "idealistic" method (*right panel*)

images of 100 males and 100 females. We transformed them into gray-scale images with resolution 64×64 (so $n = 4{,}096$).

We randomly split the 200 images into 180 for training and 20 for testing. We used the training images to compute the principal components $\boldsymbol{U}$. For each of the testing images, we randomly selected 20% of the pixels and set them to the white color ("complete corruption").

Fig. 4 "Complete corruption" scenario. The PC-projected denoised images, i.e., $\boldsymbol{U\beta}$. Different images correspond to different values of s (from small to large)

Fig. 5 "Complete corruption" scenario. The pixel-by-pixel denoised images, i.e., $\boldsymbol{x} + \boldsymbol{\alpha}$, using LAD-LASSO (*first row*), LS-LASSO (*second row*), LAD-Adaptive-LASSO (*third row*) and LS-Adaptive-LASSO (*fourth row*). Different images correspond to different values of s (from small to large)

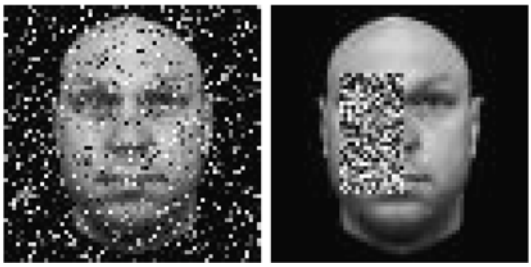

Fig. 6 *Left panel*: the corrupted pixels take values between 0 and 255 ("incomplete corruption"). *Right panel*: the corrupted pixels form a block ("block corruption")

Figure 1 in Sect. 1 illustrates an example. As we can see, with 20% corrupted pixels, the original face can hardly be recognized by visual inspection.

Figure 2 shows the first 10 principal components (or eigen-images). They are quite informative, representing different features of the faces. We used the first 100 eigen-images in our analysis.

Six different methods were compared: the ordinary least squares (OLS) (1), the "idealistic" method (Tsuda and Ratsch 2005), the LAD-LASSO (3), the LS-LASSO (4), and their adaptive versions (5) and (6). The OLS and the "idealistic" methods were used as benchmarks. In the OLS method, all pixels in the corrupted image were projected onto the PC space as in (1). The "idealistic" method first removed the corrupted pixels from the noisy image and their corresponding components from the PC images, then projected only the "good" pixels onto the PC space (via OLS). Notice that the "idealistic" method usually cannot be implemented in practice, since we do not know in advance which pixels are corrupted and which are not.

In the adaptive methods, we used the OLS result $|\beta_j^{\mathrm{OLS}}|$ as the weight for β_j, and $|x_i - \boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{\beta}^{\mathrm{OLS}}|$ as the weight for α_i.

Tuning parameters in different LASSO methods were chosen via five-fold cross-validation based on the training data (the eigen-images were also re-computed for each fold). The denoised image was compared with the original un-contaminated image using the mean squared error

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{x}_i - x_i^0)^2,$$

where $\hat{\boldsymbol{x}}$ is the denoised image, and $\boldsymbol{x}^0$ is the original un-contaminated image. Notice that each of the LASSO methods generated two denoised images: $\boldsymbol{x} + \boldsymbol{\alpha}$, the pixel by pixel denoised image, and $\boldsymbol{U}\boldsymbol{\beta}$, the PC-projected image. We compared both with the original un-contaminated image.

Numerical results are summarized in the first part of Table 1. As we can see, for this particular dataset, the OLS method performed much worse than other methods, which agrees with the general belief that some regularization is necessary. The PC-projected images $\boldsymbol{U}\boldsymbol{\beta}$ performed about the same across different methods, and they were all worse than the pixel-by-pixel denoised images. Among the pixel-by-pixel denoised images, the adaptive methods tended to work better than the non-adaptive methods. The LAD-Adaptive method also performed slightly better than the LS-Adaptive method. Interestingly, the pixel-by-pixel denoised images performed better than the "idealistic" method, by a margin as large as 40%.

Figure 3 shows examples of denoised images using the OLS method and the "idealistic" method. Figures 4 and 5 show some of the representative images along the solution paths (for different values of s) from different LASSO methods. Since the PC-projected

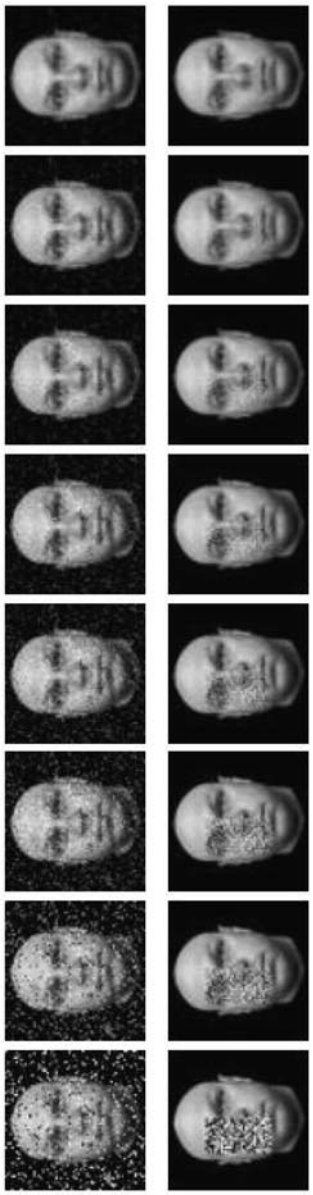

Fig. 7 The pixel-by-pixel denoised images, i.e., $\boldsymbol{x} + \boldsymbol{\alpha}$, using LAD-Adaptive-LASSO. The *first row* is for "incompletely corrupted" pixels (*left panel* in Fig. 6), and the *second row* is for "blockly corrupted" pixels (*right panel* in Fig. 6). Different images correspond to different values of s (from small to large)

images, i.e., $\boldsymbol{U\beta}$, of different models look similar, we only show one of them in Fig. 4. Figure 5 shows the pixel-by-pixel updated images, i.e., $\boldsymbol{x}+\boldsymbol{\alpha}$, corresponding to models (3)–(6). Different images correspond to different values of s (from small to large). There are several interesting patterns we can see from these figures: (1) The PC-projected images are always smooth (blurred), since they are constructed from a linear combination of PC images. (2) The LAD-based methods and the LS-based methods remove noise pixels in different ways. As s increases, the LAD-based methods keep working on a noisy pixel until it is completely recovered, before which the rest of the pixels are untouched. On the other hand the LS-based methods tend to first repair pixels with the largest noise level, so they work on different pixels "simultaneously". (3) The LAD-Adaptive-LASSO method works differently from the LAD-LASSO method. As s increases, the LAD-Adaptive-LASSO method tends to select pixels with larger noise levels and fix them first, while the LAD-LASSO method tends to randomly select a noisy pixel and fix it.

To further illustrate our methods, we also considered two other scenarios for the corrupted pixels. In the second scenario, instead of setting a corrupted pixel to the white color, i.e., "complete corruption", we set the pixel to a value uniform between 0 and 255 (left panel in Fig. 6). In the third scenario, instead of randomly selecting pixels in an image as corrupted pixels, we restricted the corrupted pixels to form a "block" (right panel in Fig. 6), and the corrupted pixels also took values uniform between 0 and 255. We note that when the corrupted pixels form a block, the L_1-norm penalty in (2) may not be the best choice, for it does not take into account the block structure. The results are summarized in the second and the third parts of Table 1 and Fig. 7. Similar as the previous result, the OLS method performed the worst for this particular dataset, and the "pixel-by-pixel" based methods performed better than the "PC-projection" based methods. Among different "pixel-by-pixel" based methods, the adaptive methods were slightly better than the non-adaptive methods. Notice that since the corrupted pixels were less "spiky" than the previous example, the least squares loss (LS) and the least absolute deviation loss (LAD) performed similarly.

5 Conclusion

In this paper, we have proposed PCA-based models that use the LASSO penalty to remove the impulse type noise for digital images. We have developed efficient solution path algorithms for these models, which facilitate the selection of the tuning parameters. We have also presented some promising evidence for our methods on a real world dataset.

Acknowledgement We would like to thank Saharon Rosset for helpful comments. Wang and Zhu are partially supported by grants DMS-0505432 and DMS-0705532 from the National Science Foundation.

References

Breiman, L. (1995). Better subset regression using the nonnegative garrote. *Technometrics*, *37*, 373–384.

Bruce, A., & Gao, H. (1996). *Applied wavelet analysis with s-plus*. New York: Springer.

Chen, S., Donoho, D., & Saunders, M. (1998). Atomic decomposition by basis pursuit. *SIAM Journal of Scientific Computing*, *20*, 33–61.

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, *32*, 407–499.

Fan, J., & Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, *96*, 1348–1360.

George, E. I., & Foster, D. P. (2000). Calibration and empirical Bayes variable selection. *Biometrika*, *87*, 731–747.

George, E. I., & McCulloch, R. E. (1993). Variable selection via Gibbs sampling. *Journal of the American Statistical Association*, *88*, 881–889.
Harwood, D., Subbararao, M., Hakalahti, H., & Davis, L. (1987). A new class of edge preserving smoothing filters. *Pattern Recognition Letters*, *6*, 155–162.
Hyvarinen, A., Hoyer, P., & Oja, E. (1998). Sparse coding shrinkage for image denoising. In *Proceedings of the IEEE international conference on neural networks* (pp. 859–864).
Mallat, S., & Zhang, Z. (1993). Matching pursuit in a time-frequency dictionary. *IEEE Transactions on Signal Processing*, *41*, 3397–3415.
Mika, S., Scholkopf, B., Smola, A., Muller, K., Scholz, M., & Ratsch, G. (1999). Kernel PCA and denoising in feature spaces. In *Advances in neural information processing systems* (pp. 537–542). Cambridge: MIT Press.
Schulze, M., & Pearce, J. (1994). A morphology-based filter structure for edge-enhancing smoothing. In *Proceedings of the IEEE international conference on image processing* (pp. 530–534).
Shang, L., Huang, D., Zheng, C., & Sun, Z. (2006). Noise removal using a novel non-negative sparse coding shrinkage technique. *Neurocomputing*, *69*, 874–877.
Shen, X., & Ye, J. (2002). Adaptive model selection. *Journal of the American Statistical Association*, *97*, 210–221.
Takahashi, T., & Kurita, T. (2002). Robust denoising by kernel PCA. In *Proceedings of the international conference on artificial neural networks* (pp. 739–744).
Thompson, A., Brown, J., Kay, J., & Titterington, D. (1991). A study of methods of choosing the smoothing parameter in image restoration by regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*, 326–339.
Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of Royal Statistical Society, Series B*, *58*, 267–288.
Troje, N., & Bulthoff, H. (1996). Face recognition under varying poses: the role of texture and shape. *Vision Research*, *36*, 1761–1771.
Tsuda, K., & Ratsch, G. (2005). Image reconstruction by linear programming. *IEEE Transactions on Image Processing*, *14*, 737–744.
Weeks, A. (1996). *Fundamentals of electronic image processing*. New York: Wiley.
Zhao, P., & Yu, B. (2006). On model selection consistency of lasso. *Journal of Machine Learning Research*, *7*, 2541–2563.
Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, *101*, 1418–1429.