CONVOLUTIONAL SPARSE REPRESENTATIONS AS AN IMAGE MODEL FOR IMPULSE NOISE RESTORATION

Brendt Wohlberg

Theoretical Division
Los Alamos National Laboratory
Los Alamos, NM 87545, USA

ABSTRACT

Standard sparse representations, applied independently to a set of overlapping image blocks, are a very effective approach to a wide variety of image reconstruction problems. Convolutional sparse representations, which provide a single-valued representation optimised over an entire image, provide an alternative form of sparse representation that has recently started to attract interest for image reconstruction problems. The present paper provides some insight into the suitability of the convolutional form for this type of application by comparing its performance as an image model with that of the standard model in an impulse noise restoration problem.

Index Terms— Sparse Representation, Convolutional Sparse Coding, Salt-and-Pepper Noise

1. INTRODUCTION

A standard sparse representation is a linear representation of the form $D\mathbf{x} \approx \mathbf{s}$, where D is the *dictionary*, $\mathbf{x}$ is the representation, and $\mathbf{s}$ is the signal to be represented. A *convolutional sparse representation* replaces this form with a sum of convolutions $\sum_m \mathbf{d}_m * \mathbf{x}_m \approx \mathbf{s}$, where the elements of the dictionary $\mathbf{d}_m$ are linear filters, and the representation consists of the set of coefficient maps $\mathbf{x}_m$, each of which is the same size as $\mathbf{s}$. This type of representation can be traced back to the *translation-invariant* sparse representations of Lewicki and Sejnowski [1], but in recent work it is typically referred to as "convolutional" rather than "translation-invariant", and is largely inspired by the *deconvolutional networks* proposed by Zeiler et al. [2] (see [3, Sec. II] for a detailed discussion of the relevant literature).

Despite the apparent advantages of this form of sparse representation, it has received little attention for image reconstruction problems: when it was first introduced, the sparse coding algorithms were too computationally expensive for such applications to be practical, and the recent resurgence of interest has largely been within the computer vision community. There are indications, however, that the image reconstruction potential of these representations is beginning to be considered, and in particular, Gu et al. have recently proposed a state-of-the-art super-resolution algorithm based on convolutional sparse representations [4]. The present paper considers impulse denoising as an example problem, but rather than attempting to construct a state-of-the-art algorithm, the focus is on exploring the different ways in which this problem can be addressed via the convolutional form (in the process proposing a number of novel approaches and extensions), and comparing with a corresponding method based on the standard form to determine whether the convolutional form offers any inherent advantages.

This research was supported by the U.S. Department of Energy via the LANL/LDRD Program, and by UC Lab Fees Research grant 12-LR-236660.

2. SPARSE CODING

The basic sparse coding problem will be posed in the form of Convolutional Basis Pursuit DeNoising (CBPDN), defined as

$$\arg\min_{\{\mathbf{x}_m\}} \frac{1}{2}\Big\|\sum_m \mathbf{d}_m * \mathbf{x}_m - \mathbf{s}\Big\|_2^2 + \lambda \sum_m \alpha_m \|\mathbf{x}_m\|_1 \ , \quad (1)$$

where the α_m allow distinct weighting of the ℓ^1 term for each filter $\mathbf{d}_m$. At present, the most efficient approach to solving this problem [3] is via the Alternating Direction Method of Multipliers (ADMM) [5] framework. An outline of the method will be presented here as a basis for extensions proposed in following sections.

Problem (1) can be written as

$$\arg\min_{\mathbf{x}} \frac{1}{2}\big\|D\mathbf{x} - \mathbf{s}\big\|_2^2 + \lambda \|\boldsymbol{\alpha} \odot \mathbf{x}\|_1 \ , \quad (2)$$

where D_m is a linear operator such that $D_m\mathbf{x}_m = \mathbf{d}_m * \mathbf{x}_m$, and D, $\boldsymbol{\alpha}$, and $\mathbf{x}$ are the block matrices/vectors

$$D = \begin{pmatrix} D_0 & D_1 & \dots \end{pmatrix} \quad \boldsymbol{\alpha} = \begin{pmatrix} \alpha_0\mathbf{1} \\ \alpha_1\mathbf{1} \\ \vdots \end{pmatrix} \quad \mathbf{1} = \begin{pmatrix} 1 \\ 1 \\ \vdots \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \vdots \end{pmatrix} \ . \quad (3)$$

The simplest variable splitting into the ADMM standard form is

$$\arg\min_{\mathbf{x},\mathbf{y}} \frac{1}{2}\big\|D\mathbf{x} - \mathbf{s}\big\|_2^2 + \lambda \|\boldsymbol{\alpha} \odot \mathbf{y}\|_1 \text{ s.t. } \mathbf{x} - \mathbf{y} = 0 \ , \quad (4)$$

for which the corresponding ADMM iterations are

$$\mathbf{x}^{(j+1)} = \arg\min_{\mathbf{x}} \frac{1}{2}\big\|D\mathbf{x} - \mathbf{s}\big\|_2^2 + \frac{\rho}{2}\Big\|\mathbf{x} - \mathbf{y}^{(j)} + \mathbf{u}^{(j)}\Big\|_2^2 \quad (5)$$

$$\mathbf{y}^{(j+1)} = \arg\min_{\mathbf{y}} \lambda \|\boldsymbol{\alpha} \odot \mathbf{y}\|_1 + \frac{\rho}{2}\Big\|\mathbf{x}^{(j+1)} - \mathbf{y} + \mathbf{u}^{(j)}\Big\|_2^2 \quad (6)$$

$$\mathbf{u}^{(j+1)} = \mathbf{u}^{(j)} + \mathbf{x}^{(j+1)} - \mathbf{y}^{(j+1)} \ . \quad (7)$$

The solution to (6) is given by

$$\mathbf{y}_m^{(j+1)} = \mathcal{S}_{\alpha_m\lambda/\rho}\Big(\mathbf{x}_m^{(j+1)} + \mathbf{u}_m^{(j)}\Big) \ , \quad (8)$$

where

$$\mathcal{S}_\gamma(\mathbf{u}) = \text{sign}(\mathbf{u}) \odot \max(0, |\mathbf{u}| - \gamma) \ . \quad (9)$$

The only computationally expensive step is (5), which can be solved via the equivalent DFT domain problem

$$\arg\min_{\hat{\mathbf{x}}} \frac{1}{2}\big\|\hat{D}\hat{\mathbf{x}} - \hat{\mathbf{s}}\big\|_2^2 + \frac{\rho}{2}\|\hat{\mathbf{x}} - (\hat{\mathbf{y}} - \hat{\mathbf{u}})\|_2^2 \ , \quad (10)$$

where $\hat{D} = \begin{pmatrix} \hat{D}_0 & \hat{D}_1 & \dots \end{pmatrix}$ and

$$\hat{\mathbf{x}} = \begin{pmatrix} \hat{\mathbf{x}}_0 \\ \hat{\mathbf{x}}_1 \\ \vdots \end{pmatrix} \quad \hat{\mathbf{y}} = \begin{pmatrix} \hat{\mathbf{y}}_0 \\ \hat{\mathbf{y}}_1 \\ \vdots \end{pmatrix} \quad \hat{\mathbf{u}} = \begin{pmatrix} \hat{\mathbf{u}}_0 \\ \hat{\mathbf{u}}_1 \\ \vdots \end{pmatrix} \; . \tag{11}$$

The solution for (10) is given by the $MN \times MN$ (for M filters and an image $\mathbf{s}$ with N pixels) linear system

$$(\hat{D}^H \hat{D} + \rho I)\hat{\mathbf{x}} = \hat{D}^H \hat{\mathbf{s}} + \rho\,(\hat{\mathbf{y}} - \hat{\mathbf{u}}) \; . \tag{12}$$

The key to solving this very large linear system is the observation that it can be decomposed into N independent $M \times M$ linear systems [6], each of which has a system matrix consisting of the sum of rank-one and diagonal terms so they they can be solved very efficiently by exploiting the Sherman-Morrison formula [7].

3. LOWPASS COMPONENT REPRESENTATION

Since convolutional sparse representations do not provide a good representation of the low-frequency components of an image, it is common practice to compute the representation from a contrast-normalized [8] or highpass filtered [3] version of the image to be decomposed. Pre-processing by linear filtering in the presence of impulse noise does not provide a good estimate of the image lowpass component, so an alternative is necessary. The natural solution is to explicitly include a lowpass component in the representation, which can be achieved in a few different ways.

The simplest, not requiring any modification to the form of (1), is to append a Gaussian filter of large support to the dictionary, setting the corresponding weight α_m to zero to ensure that the ℓ^1 regularization does not introduce artifacts into the low-frequency representation. This approach is quite workable, but the large filter support requires more computational resources to be expended on boundary handling, and care has to be taken to ensure that the Gaussian has decayed sufficiently at the filter boundaries to avoid potentially severe artifacts in the lowpass component.

An alternative is to introduce an impulse filter, again setting the corresponding α_m to zero, together with an additional regularization term that can impose smoothness on the corresponding coefficient map. We will see in the following section that it is possible to introduce such a regularization term while retaining the efficient solution method outlined in Sec. 2.

4. GRADIENT REGULARIZATION

An extension of (1) to include regularization on the gradients of the coefficient maps can be defined as

$$\arg\min_{\{\mathbf{x}_m\}} \frac{1}{2} \left\| \sum_m \mathbf{d}_m * \mathbf{x}_m - \mathbf{s} \right\|_2^2 + \lambda \sum_m \alpha_m \left\| \mathbf{x}_m \right\|_1 + \frac{\mu}{2} \sum_m \beta_m \left\| \sqrt{(\mathbf{g}_0 * \mathbf{x}_m)^2 + (\mathbf{g}_1 * \mathbf{x}_m)^2} \right\|_2^2 \; , \tag{13}$$

where $\mathbf{g}_0$ and $\mathbf{g}_1$ are filters that compute the gradients along image rows and columns respectively. The final term can be written as $\frac{\mu}{2} \sum_m \beta_m \left\| G_0 \mathbf{x}_m \right\|_2^2 + \frac{\mu}{2} \sum_m \beta_m \left\| G_1 \mathbf{x}_m \right\|_2^2$, where linear operators G_0 and G_1 are defined such that $G_l \mathbf{x}_m = \mathbf{g}_l * \mathbf{x}_m$. Introducing block matrix notation as before, (13) can be expressed as

$$\arg\min_{\mathbf{x}} \frac{1}{2} \left\| D\mathbf{x} - \mathbf{s} \right\|_2^2 + \lambda \left\| \boldsymbol{\alpha} \odot \mathbf{x} \right\|_1 + \frac{\mu}{2} \left\| \Gamma_0 \mathbf{x} \right\|_2^2 + \frac{\mu}{2} \left\| \Gamma_1 \mathbf{x} \right\|_2^2 \; , \tag{14}$$

where

$$\Gamma_l = \begin{pmatrix} \sqrt{\beta_0} G_l & 0 & \dots \\ 0 & \sqrt{\beta_1} G_l & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \; . \tag{15}$$

Since the gradient terms consist of squared ℓ^2 norms, the same splitting into standard ADMM form as in (4) can be applied, substituting $\mathbf{y}$ into the ℓ^1 term, and grouping together the remaining squared ℓ^2 terms in $\mathbf{x}$. The resulting $\mathbf{x}$ subproblem corresponding to (5) has the form

$$\arg\min_{\mathbf{x}} \frac{1}{2} \left\| D\mathbf{x} - \mathbf{s} \right\|_2^2 + \frac{\mu}{2} \left\| \Gamma_0 \mathbf{x} \right\|_2^2 + \frac{\mu}{2} \left\| \Gamma_1 \mathbf{x} \right\|_2^2 + \frac{\rho}{2} \left\| \mathbf{x} - \mathbf{y} + \mathbf{u} \right\|_2^2 \; . \tag{16}$$

The solution of the equivalent DFT domain problem is given by

$$(\hat{D}^H \hat{D} + \mu \hat{\Gamma}_0^H \hat{\Gamma}_0 + \mu \hat{\Gamma}_1^H \hat{\Gamma}_1 + \rho I)\hat{\mathbf{x}} = \hat{D}^H \hat{\mathbf{s}} + \rho(\hat{\mathbf{y}} - \hat{\mathbf{u}}) \; . \tag{17}$$

Since $\hat{\Gamma}_0^H \hat{\Gamma}_0$ and $\hat{\Gamma}_1^H \hat{\Gamma}_1$ are diagonal they can be grouped together with the ρI term; the independent linear systems described in Sec. 2 are again composed from rank-one and diagonal terms and the Sherman-Morrison solution can be directly applied without any substantial increase in computational cost.

5. IMPULSE NOISE RESTORATION

Denoising of salt-and-pepper noise – a type of impulse noise that corresponds to a randomly distributed set of pixels being set to either the minimum or maximum pixel value – is a convenient example restoration problem for an exploration of the restoration performance of convolutional sparse representations. A hybrid of adaptive median filtering and ℓ^1-Total Variation has shown very good performance in this problem [9]. More recently, a number of authors have proposed solutions based on standard sparse representations [10, 11, 12, 13]. Most of the differences between these methods can be decomposed into a few independent features:

Dictionary A fixed DCT dictionary is used in [10], while the other three methods learn a data-adaptive dictionary as part of the restoration algorithm [11, 12, 13].

Noise Detection The methods in [10, 12, 13] include an explicit impulse noise detection stage, while [12] does not.

Sparse Coding The methods in [10, 13] use an ℓ^2 data fidelity term, while [11, 12] use an ℓ^1 data fidelity term.

(The method of [13] is somewhat more complex than the others, and is not adequately summarised by these primary features.) A feature shared by all of these methods is the use of weighted averaging to aggregate the independent pixel estimates from overlapping patches.

Since the primary purpose of this work is to compare standard and convolutional sparse representations, a very simple restoration framework is considered here. In particular, there is no explicit noise detection phase, and fixed dictionaries learned on a separate data set are used, with no adaptation to the image being restored. Instead of using an ℓ^1 data fidelity term to avoid the need for explicit noise detection, an augmented dictionary is used to allow use of the more common Basis Pursuit DeNoising (BPDN) problem

$$\arg\min_{\mathbf{x}} \frac{1}{2} \left\| D\mathbf{x} - \mathbf{s} \right\|_2^2 + \lambda \left\| \boldsymbol{\alpha} \odot \mathbf{x} \right\|_1 \; , \tag{18}$$

which is applied independently to a set of overlapping image blocks extracted from the image using a fixed step size between each consecutive block. The actual dictionary D used in each decomposition

consists of the concatenation of a learned dictionary D_0, a constant column vector to represent the block mean (assigned a zero weight α in the regularization term), and an identity matrix I, the columns of which are able to implicitly detect and represent the impulse noise. The weighting vector $\boldsymbol{\alpha}$ enables a different weight to be given to this impulse dictionary, to be varied according to the amount of impulse noise. The denoised estimate for each block is obtained by excluding this dictionary component and the corresponding coefficients in $\mathbf{x}$ when reconstructing the representation. If the step between blocks is a single pixel then the representation is shift-invariant like the convolutional form, but without joint optimization over all the blocks so that there are multiple estimates for each pixel. How to aggregate these estimates is a critical issue. Although previous methods based on sparse representations of overlapping blocks [11, 12, 13] have aggregated by averaging, a more robust estimator such as the median is an obvious alternative given the non-Gaussian nature of the noise.

The corresponding convolutional form is conceptually the same, the major differences being that there is no need for an aggregation stage, and the dictionary is simply augmented with a single impulse filter to represent the impulse noise.

6. CBPDN WITH ℓ^1 DATA FIDELITY TERM

Since most standard sparse representation methods for impulse noise restoration without an explicit noise detection stage employ an ℓ^1 data fidelity term, it is worth exploring a convolutional variant, which can be written (using the notation defined in Sec. 2) as

$$\arg\min_{\mathbf{x}} \left\| D\mathbf{x} - \mathbf{s} \right\|_1 + \lambda \left\| \boldsymbol{\alpha} \odot \mathbf{x} \right\|_1 \ . \tag{19}$$

This problem can again be expressed in standard ADMM form

$$\begin{aligned} &\arg\min_{\mathbf{x},\mathbf{y}_0,\mathbf{y}_1} \left\| \mathbf{y}_1 \right\|_1 + \lambda \left\| \boldsymbol{\alpha} \odot \mathbf{y}_0 \right\|_1 \\ &\text{s.t.} \ \begin{pmatrix} \mathbf{x} \\ D\mathbf{x} \end{pmatrix} - \begin{pmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \end{pmatrix} = \begin{pmatrix} 0 \\ \mathbf{s} \end{pmatrix} \ . \end{aligned} \tag{20}$$

by employing a different variable splitting from that used in (4). Defining

$$A = \begin{pmatrix} I \\ D \end{pmatrix} \quad \mathbf{y} = \begin{pmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \end{pmatrix} \quad \mathbf{u} = \begin{pmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \end{pmatrix} \quad \mathbf{c} = \begin{pmatrix} 0 \\ \mathbf{s} \end{pmatrix} \ , \tag{21}$$

the corresponding ADMM iterations are

$$\mathbf{x}^{(j+1)} = \arg\min_{\mathbf{x}} \frac{\rho}{2} \left\| A\mathbf{x} - \mathbf{y}^{(j)} - \mathbf{c} + \mathbf{u}^{(j)} \right\|_2^2 \tag{22}$$

$$\begin{aligned} \mathbf{y}^{(j+1)} = \arg\min_{\mathbf{y}} &\left\| \mathbf{y}_1 \right\|_1 + \lambda \left\| \boldsymbol{\alpha} \odot \mathbf{y}_0 \right\|_1 + \\ &\frac{\rho}{2} \left\| A\mathbf{x}^{(j+1)} - \mathbf{y} - \mathbf{c} + \mathbf{u}^{(j)} \right\|_2^2 \end{aligned} \tag{23}$$

$$\mathbf{u}^{(j+1)} = \mathbf{u}^{(j)} + A\mathbf{x}^{(j+1)} - \mathbf{y}^{(j+1)} - \mathbf{c} \ . \tag{24}$$

The functional minimised in (22) can be expanded as

$$\begin{aligned} \frac{\rho}{2} \left\| A\mathbf{x} - \mathbf{y} - \mathbf{c} + \mathbf{u} \right\|_2^2 = &\frac{\rho}{2} \left\| D\mathbf{x} - (\mathbf{y}_1 + \mathbf{s} - \mathbf{u}_1) \right\|_2^2 + \\ &\frac{\rho}{2} \left\| \mathbf{x} - (\mathbf{y}_0 - \mathbf{u}_0) \right\|_2^2 \ , \end{aligned} \tag{25}$$

which is of the same form as (5), and can be solved via the same frequency domain method. The functional minimised in (23) can be expanded as

$$\begin{aligned} \left\| \mathbf{y}_1 \right\|_1 + \lambda \left\| \boldsymbol{\alpha} \odot \mathbf{y}_0 \right\|_1 + \frac{\rho}{2} \left\| \mathbf{y}_1 - (D\mathbf{x} - \mathbf{s} + \mathbf{u}_1) \right\|_2^2 \\ + \frac{\rho}{2} \left\| \mathbf{y}_0 - (\mathbf{x} + \mathbf{u}_0) \right\|_2^2 \ . \end{aligned} \tag{26}$$

Since the $\mathbf{y}_0$ and $\mathbf{y}_1$ components of $\mathbf{y}$ are decoupled, minimisation with respect to $\mathbf{y}$ can be achieved by the independent minimisations

$$\mathbf{y}_0^{(j+1)} = \arg\min_{\mathbf{y}_0} \lambda \left\| \boldsymbol{\alpha} \odot \mathbf{y}_0 \right\|_1 + \frac{\rho}{2} \left\| \mathbf{y}_0 - (\mathbf{x} + \mathbf{u}_0) \right\|_2^2 \tag{27}$$

$$\mathbf{y}_1^{(j+1)} = \arg\min_{\mathbf{y}_1} \left\| \mathbf{y}_1 \right\|_1 + \frac{\rho}{2} \left\| \mathbf{y}_1 - (D\mathbf{x} - \mathbf{s} + \mathbf{u}_1) \right\|_2^2 . \tag{28}$$

The solution to both of these problems is given by soft thresholding as in (8),(9).

The additional gradient regularisation term described in Sec. 4 is also easily incorporated, leading to the problem

$$\arg\min_{\mathbf{x}} \left\| D\mathbf{x} - \mathbf{s} \right\|_1 + \lambda \left\| \boldsymbol{\alpha} \odot \mathbf{x} \right\|_1 + \frac{\mu}{2} \left\| \Gamma_0 \mathbf{x} \right\|_2^2 + \frac{\mu}{2} \left\| \Gamma_1 \mathbf{x} \right\|_2^2 \ . \tag{29}$$

The only modification necessary to the ADMM algorithm above is in the $\mathbf{x}$ update, which has the form

$$\begin{aligned} \frac{\rho}{2} \left\| D\mathbf{x} - (\mathbf{y}_1 + \mathbf{s} - \mathbf{u}_1) \right\|_2^2 + \frac{\rho}{2} \left\| \mathbf{x} - (\mathbf{y}_0 - \mathbf{u}_0) \right\|_2^2 \\ + \frac{\mu}{2} \left\| \Gamma_0 \mathbf{x} \right\|_2^2 + \frac{\mu}{2} \left\| \Gamma_1 \mathbf{x} \right\|_2^2 \ . \end{aligned} \tag{30}$$

It is also worth noting that it is trivial to introduce a spatial weighting matrix into this ℓ^1 data fidelity term, as in the mask decoupling form of CBPDN [14, 15].

7. RESULTS

(a) Reference (b) Corrupted

Fig. 1. Ground truth and corrupted test images.

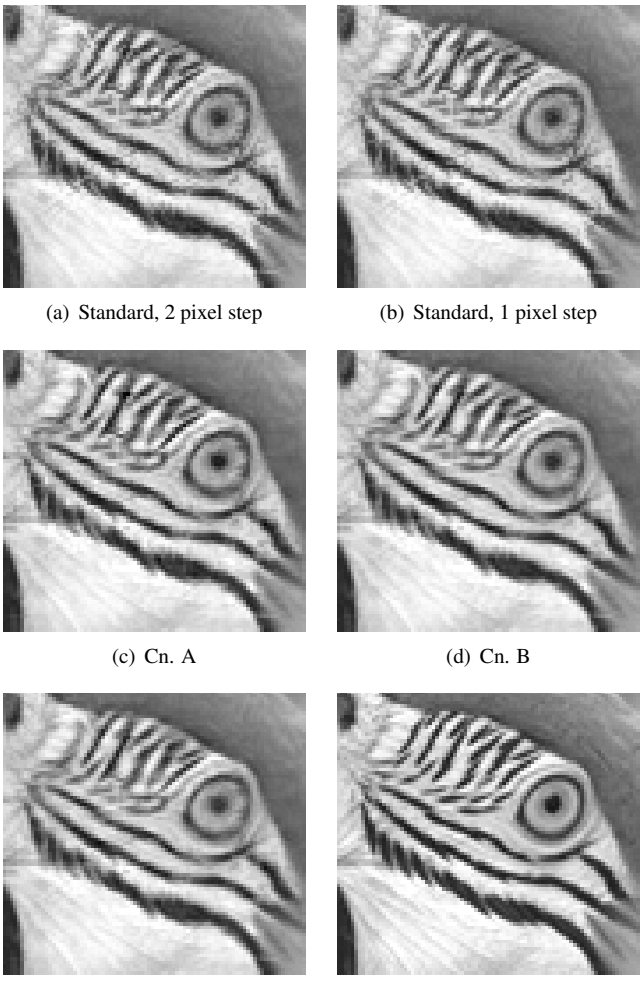
A 360×360 pixel crop from the "kodim23" standard image [16] was used as a ground truth image (see Fig. 1(a)). The corresponding test image (see Fig. 1(b)) was generated by corrupting it with 30% salt-and-pepper noise (i.e. 30% of the pixels, at randomly selected locations were set at random to either 0 or 255). The test image was restored using the methods based on standard and convolutional sparse representations described in Sec. 5. For a fair comparison, each method made use of its own dictionary, both learned from the same set of seven 512×512 pixel training images, but with the appropriate dictionary learning algorithms[1].

The results for the standard sparse coding method with blocks overlapped with steps of 4, 2, and 1 pixels and median aggregation

[1] In order to ensure that this did not inadvertently confer an advantage on one method – for example, one might suppose that the shift-invariant convolutional dictionary could be inherently superior – each method was cross-tested with the dictionary learned for the other method. In both cases this led to inferior performance than obtained using the "correct" dictionary.

	St. 4	St. 2	St. 1	Cn. A	Cn. B	Cn. C
PSNR	31.13	31.66	31.75	32.08	32.51	32.41
SSIM	0.937	0.942	0.943	0.941	0.941	0.940

Table 1. Restoration performance of different sparse coding methods. St. $<n>$ represent standard sparse coding with blocks overlapped with an n pixel step and with median aggregation. Cn. A, B, and C represent convolutional methods as described in the main text.



(a) Standard, 2 pixel step (b) Standard, 1 pixel step

(c) Cn. A (d) Cn. B

(e) Cn. C (f) Reference

Fig. 2. A crop of a detailed region in the reference and reconstructed images. Cn. A, B, and C represent convolutional methods as described in the main text.

are displayed in the first three columns of Table 1. The corresponding results for aggregation by weighted averaging were substantially inferior, with PSNR values of 30.51dB, 30.86dB, and 30.92dB for 4, 2, and 1 pixel steps respectively. The results for the convolutional sparse coding methods are displayed in the last three columns of Table 1. For the first of these, Cn. A in Table 1, all β_m in (13) except for that corresponding to the impulse filter for representing the image lowpass component were set to zero, so that the only role of the gradient regularization was in representing that component. For Cn. B in Table 1 all of the β_m that were zero for Cn. A were set to the same non-zero value, so that the gradient regularization had an effect on the main part of the sparse representation as well. The method Cn. C in Table 1 used full gradient regularization as in Cn. B, but with the ℓ^2 data fidelity and dedicated impulse filter for representing the impulse noise replaced with an ℓ^1 data fidelity. (As in the case of the ℓ^2 data fidelity, setting most of the β_m to zero results in an inferior reconstruction PSNR.) In all cases the parameters were manually selected to optimise PSNR results.

The convolutional methods provide better performance than the standard methods in terms of PSNR, but the SSIM values differ by negligible amounts. A plausible explanation for this phenomenon is that the only advantage of the convolutional methods is in providing a superior reconstruction of the lowpass image component, which is consistent with the observation that estimation of the block mean can be problematic for standard sparse coding methods [12, Sec. 4.2]. However, examination of the examples in Fig. 2 reveals that the convolutional methods are superior in terms of subjective image quality, with fewer visually objectionable artifacts and without any corresponding reduction in edge sharpness. It is also worth noting that the convolutional methods with additional gradient regularization (Cn. B and Cn. C) provide a higher PSNR than that with only the usual ℓ^1 regularization; although the subjective quality differences are very small, there are some contexts, such as the reconstruction of scientific data, in which the PSNR is more relevant than subjective quality. The differences between the convolutional methods using an ℓ^2 data fidelity with augmented dictionary and using an ℓ^1 data fidelity are negligible, both in terms of objective metrics and subjective quality.

8. CONCLUSIONS

One should not assume that the optimisation of the representation over the entire image automatically confers a major advantage on convolutional sparse representations. If the independent estimates arising from each overlapping block in a standard sparse representations approach are simply averaged, then the convolutional form enjoys a substantial advantage, but if the aggregation method is chosen more carefully, the standard approach is much more competitive. Nevertheless, at least in the application considered here, the convolutional form does appear to enjoy a small but significant advantage in terms of both reconstruction PSNR and subjective image quality.

With respect to the specific ways of addressing the problem via the convolutional form, it is notable that (i) the basic CBPDN problem can be augmented with a gradient regularization term while retaining the computationally efficient DFT domain solution, and this additional regularization provides a convenient method for representing the image lowpass component, and also confers at least some additional advantage in terms of reconstruction quality, (ii) it is also possible to retain the efficient DFT domain solution in solving a variant of this problem with an ℓ^1 data fidelity term, but performance comparisons indicate that the ℓ^1 data fidelity term approach has no inherent advantage over an ℓ^2 data fidelity term with an augmented dictionary to represent impulses.

Implementations of the algorithms proposed here will be included in a future release of the SPORCO library [17].

9. ACKNOWLEDGMENT

The author thanks A. Foi for insightful discussion on the aggregation of independent estimates in image reconstruction problems.

10. REFERENCES

[1] M. S. Lewicki and T. J. Sejnowski, "Coding time-varying signals using sparse, shift-invariant representations," in *Advances in Neural Information Processing Systems 11*, 1999, pp. 730–736.

[2] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *Proc. IEEE Conf. Comp. Vis. Pat. Recog. (CVPR)*, Jun. 2010, pp. 2528–2535. doi:10.1109/cvpr.2010.5539957

[3] B. Wohlberg, "Efficient algorithms for convolutional sparse representations," *IEEE Trans. Image Process.*, vol. 25, no. 1, pp. 301–315, Jan. 2016. doi:10.1109/TIP.2015.2495260

[4] S. Gu, W. Zuo, Q. Xie, D. Meng, X. Feng, and L. Zhang, "Convolutional sparse coding for image super-resolution," in *Proc. IEEE Int. Conf. Comp. Vis. (ICCV)*, Dec. 2015. doi:10.1109/ICCV.2015.212

[5] 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*, vol. 3, no. 1, pp. 1–122, 2010. doi:10.1561/2200000016

[6] H. Bristow, A. Eriksson, and S. Lucey, "Fast convolutional sparse coding," in *Proc. IEEE Conf. Comp. Vis. Pat. Recog. (CVPR)*, Jun. 2013, pp. 391–398. doi:10.1109/CVPR.2013.57

[7] B. Wohlberg, "Efficient convolutional sparse coding," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, May 2014, pp. 7173–7177. doi:10.1109/ICASSP.2014.6854992

[8] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *Proc. IEEE Int. Conf. Comp. Vis. (ICCV)*, Nov. 2011, pp. 2018–2025. doi:10.1109/iccv.2011.6126474

[9] R. H. Chan, C.-W. Ho, and M. Nikolova, "Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization," *IEEE Trans. Image Process.*, vol. 14, no. 10, pp. 1479–1485, Oct. 2005. doi:10.1109/tip.2005.852196

[10] P. Saikrishna and P. K. Bora, "Detection and removal of random-valued impulse noise from images using sparse representations," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2013, pp. 1197–1201. doi:10.1109/icip.2013.6738247

[11] F. Chen, G. Ma, L. Lin, and Q. Qin, "Impulsive noise removal via sparse representation," *Journal of Electronic Imaging*, vol. 22, no. 4, p. 043014, Nov. 2013. doi:10.1117/1.jei.22.4.043014

[12] S. Wang, Q. Liu, Y. Xia, P. Dong, J. Luo, Q. Huang, and D. D. Feng, "Dictionary learning based impulse noise removal via L1-L1 minimization," *Signal Processing*, vol. 93, no. 9, pp. 2696–2708, Sep. 2013. doi:10.1016/j.sigpro.2013.03.005

[13] C. L. P. Chen, L. Liu, L. Chen, Y. Y. Tang, and Y. Zhou, "Weighted couple sparse representation with classified regularization for impulse noise removal," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 4014–4026, Nov. 2015. doi:10.1109/tip.2015.2456432

[14] F. Heide, W. Heidrich, and G. Wetzstein, "Fast and flexible convolutional sparse coding," in *Proc. IEEE Conf. Comp. Vis. Pat. Recog. (CVPR)*, 2015, pp. 5135–5143. doi:10.1109/CVPR.2015.7299149

[15] B. Wohlberg, "Boundary handling for convolutional sparse representations," in *Proc. IEEE Conf. Image Process. (ICIP)*, Sep. 2016, pp. 1833–1837. doi:10.1109/ICIP.2016.7532675

[16] "Kodak lossless true color image suite," Available from http://r0k.us/graphics/kodak/.

[17] B. Wohlberg, "SParse Optimization Research COde (SPORCO)," Software library available from http://purl.org/brendt/software/sporco, 2016, version (Matlab) 0.0.3.