CONVOLUTIONAL SPARSE REPRESENTATIONS WITH GRADIENT PENALTIES

Brendt Wohlberg

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

ABSTRACT

While convolutional sparse representations enjoy a number of useful properties, they have received limited attention for image reconstruction problems. The present paper compares the performance of block-based and convolutional sparse representations in the removal of Gaussian white noise. The usual formulation of the convolutional sparse coding problem is slightly inferior to the block-based representations in this problem, but the performance of the convolutional form can be boosted beyond that of the block-based form by the inclusion of suitable penalties on the gradients of the coefficient maps.

***Index Terms*—** Convolutional Sparse Representations, Convolutional Sparse Coding, Total Variation

1. INTRODUCTION

Sparse representations are well-established as a tool for inverse problems in a wide variety of areas, including signal and image processing, computer vision, and machine learning [1]. The standard form is a linear representation $D\mathbf{x} \approx \mathbf{s}$, where D is the *dictionary*, $\mathbf{x}$ is the representation, and $\mathbf{s}$ is the signal to be represented. When D is a linear transform with a fast transform operator, such as the Discrete Wavelet Transform, these representations can be computed for large images, but when D is learned from training data and represented as an explicit matrix, this is not feasible, the standard approach being to independently compute the representations over a set of overlapping image patches. *Convolutional sparse representations* are a recent[1] alternative that replace the general linear representation with a sum of convolutions[2] $\sum_m \mathbf{d}_m * \mathbf{x}_m \approx \mathbf{s}$, where the elements of the dictionary $\mathbf{d}_m$ are linear filters, and the representation consists of the set of coefficient maps $\mathbf{x}_m$.

There is growing interest in imaging and image processing applications of the convolutional form [4, 5, 6, 7, 8, 9]. Surprisingly, denoising of Gaussian white noise, arguably the simplest of all imaging inverse problems, has received almost no attention beyond a very brief example providing insufficient detail for reproducibilty [10, Sec. 4.4]. The present paper argues that, despite its numerous advantages in many contexts, the convolutional form is not competitive for the Gaussian white noise denoising problem, but that these deficiencies can be mitigated by moving beyond simple ℓ_1 regularization, the specific form being investigated here consisting of additional penalties on the gradients of the coefficient map[3].

It is emphasised that these extensions have relevance beyond the specific denoising test problem considered here, in that the improved performance reported on this problem can also be expected to have an impact on more general image reconstruction problems, e.g. when convolutional sparse coding is employed as the prior within the plug-and-play priors framework [12, 13]. There is also evidence that the inclusion of such gradient penalties enhances the performance of convolutional sparse representations in certain image decomposition/restoration problems [7, 9].

2. CONVOLUTIONAL SPARSE CODING

The most widely used form of convolutional sparse coding is Convolutional Basis Pursuit DeNoising (CBPDN), defined as

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

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 [2] is via the Alternating Direction Method of Multipliers (ADMM) [14] framework. An outline of this method is presented here as a basis for extensions proposed in following sections.

Problem (1) can be written as

$$\operatorname*{arg\,min}_{\mathbf{x}} \;(1/2)\big\|D\mathbf{x} - \mathbf{s}\big\|_2^2 + \lambda \left\|\boldsymbol{\alpha} \odot \mathbf{x}\right\|_1 \;, \quad (2)$$

where $\odot$ is the Hadamard product, 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

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

This problem can be expressed in ADMM standard form as

$$\operatorname*{arg\,min}_{\mathbf{x},\mathbf{y}} \;(1/2)\big\|D\mathbf{x} - \mathbf{s}\big\|_2^2 + \lambda \left\|\boldsymbol{\alpha} \odot \mathbf{y}\right\|_1 \;\text{s.t.}\; \mathbf{x} - \mathbf{y} = 0 \;, \quad (4)$$

which can be solved via the ADMM iterations

$$\mathbf{x}^{(j+1)} = \operatorname*{arg\,min}_{\mathbf{x}} \frac{1}{2}\big\|D\mathbf{x} - \mathbf{s}\big\|_2^2 + \frac{\rho}{2}\left\|\mathbf{x} - \mathbf{y}^{(j)} + \mathbf{u}^{(j)}\right\|_2^2 \quad (5)$$

$$\mathbf{y}^{(j+1)} = \operatorname*{arg\,min}_{\mathbf{y}} \lambda \left\|\boldsymbol{\alpha} \odot \mathbf{y}\right\|_1 + \frac{\rho}{2}\left\|\mathbf{x}^{(j+1)} - \mathbf{y} + \mathbf{u}^{(j)}\right\|_2^2 \quad (6)$$

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

The solution to (6) is given by the soft thresholding operation [15, Sec. 6.5.2] $\mathbf{y} = \operatorname{sign}(\mathbf{z}) \odot \max(0, |\mathbf{z}| - \lambda\boldsymbol{\alpha}/\rho)$ where $\mathbf{z} = \mathbf{x} + \mathbf{u}$. The only computationally expensive step is (5), which can be solved via the equivalent DFT domain problem

This research was supported by the U.S. Department of Energy via the LANL/LDRD Program.

[1]More accurately, the label *convolutional* is recent, but the equivalent *translation invariant sparse representations* are much older [2, Sec. II].

[2]Typically circular convolutions [3].

[3]A weighting strategy applied to the ℓ_1 penalty has also been found to improve the denoising performance of convolutional sparse representations [11, Sec. 8], but that approach is not considered here due to space constraints.

$$\operatorname*{arg\,min}_{\hat{\mathbf{x}}} \; (1/2)\left\|\hat{D}\hat{\mathbf{x}} - \hat{\mathbf{s}}\right\|_2^2 + (\rho/2) \left\|\hat{\mathbf{x}} - (\hat{\mathbf{y}} - \hat{\mathbf{u}})\right\|_2^2 \;, \tag{8}$$

where $\hat{\mathbf{z}}$ denotes the DFT of variable $\mathbf{z}$. The solution for (8) is given by the $MN \times MN$ linear system (for M filters and an image $\mathbf{s}$ with N pixels)

$$(\hat{D}^H \hat{D} + \rho I)\hat{\mathbf{x}} = \hat{D}^H \hat{\mathbf{s}} + \rho \left(\hat{\mathbf{y}} - \hat{\mathbf{u}}\right) \;. \tag{9}$$

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 [16], 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 [17].

3. GRADIENT REGULARIZATION

An extension of (1) to include an ℓ_2 penalty on the gradients of the coefficient maps was proposed in [6]. The primary purpose of this extension was as a regularization for an impulse filter intended to represent the low-frequency components of the image, but a small non-zero regularization on the other dictionary filters was found to provide a small improvement to the impulse noise denoising performance [6]. Considering the edge-smoothing effect of ℓ_2 gradient regularization, a reasonable alternative to consider is Total Variation (TV) regularization. We consider three different variants:

1. scalar TV [18] applied independently to each coefficient map,
2. vector TV [19] applied jointly to the set of coefficient maps,
3. scalar TV [18] applied to the reconstructed image components $D_m \mathbf{x}_m$ rather than to the coefficient maps $\mathbf{x}_m$.

3.1. Scalar TV on Coefficient Map

The CBPDN problem extended by adding a scalar TV term on each coefficient map can be written as

$$\operatorname*{arg\,min}_{\{\mathbf{x}_m\}} \frac{1}{2}\left\|\sum_m \mathbf{d}_m * \mathbf{x}_m - \mathbf{s}\right\|_2^2 + \lambda \sum_m \alpha_m \left\|\mathbf{x}_m\right\|_1 + \mu \sum_m \beta_m \left\|\sqrt{(\mathbf{g}_0 * \mathbf{x}_m)^2 + (\mathbf{g}_1 * \mathbf{x}_m)^2}\right\|_1 \;, \tag{10}$$

where $\mathbf{g}_0$ and $\mathbf{g}_1$ are filters that compute the gradients along image rows and columns respectively. The TV term can be written as $\mu \sum_m \beta_m \left\|\sqrt{(G_0\mathbf{x}_m)^2 + (G_1\mathbf{x}_m)^2}\right\|_1$ where linear operators G_0 and G_1 are defined such that $G_l \mathbf{x}_m = \mathbf{g}_l * \mathbf{x}_m$, and defining[4]

$$\Gamma_l = \begin{pmatrix} \beta_0 G_l & 0 & \dots \\ 0 & \beta_1 G_l & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \tag{11}$$

allows further reduction to $\mu \left\|\sqrt{(\Gamma_0\mathbf{x})^2 + (\Gamma_1\mathbf{x})^2}\right\|_1$.

Problem (10) can be written in standard ADMM form as

$$\operatorname*{arg\,min}_{\mathbf{x},\mathbf{y}_0,\mathbf{y}_1,\mathbf{y}_2} \frac{1}{2}\left\|D\mathbf{x} - \mathbf{s}\right\|_2^2 + \lambda \left\|\boldsymbol{\alpha} \odot \mathbf{y}_2\right\|_1 + \mu \left\|\sqrt{\mathbf{y}_0^2 + \mathbf{y}_1^2}\right\|_1 \quad \text{s.t.} \begin{pmatrix} \Gamma_0 \mathbf{x} \\ \Gamma_1 \mathbf{x} \\ \mathbf{x} \end{pmatrix} - \begin{pmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = 0 \;. \tag{12}$$

The resulting $\mathbf{x}$ subproblem has the form

$$\operatorname*{arg\,min}_{\mathbf{x}} \frac{1}{2}\left\|D\mathbf{x} - \mathbf{s}\right\|_2^2 + \frac{\rho}{2}\left\|\Gamma_0\mathbf{x} - \mathbf{y}_0 + \mathbf{u}_0\right\|_2^2 + \frac{\rho}{2}\left\|\Gamma_1\mathbf{x} - \mathbf{y}_1 + \mathbf{u}_1\right\|_2^2 + \frac{\rho}{2}\left\|\mathbf{x} - \mathbf{y}_2 + \mathbf{u}_2\right\|_2^2 \;, \tag{13}$$

and the solution of the equivalent DFT domain problem is given by

$$(\hat{D}^H \hat{D} + \rho I + \rho \hat{\Gamma}_0^H \hat{\Gamma}_0 + \rho \hat{\Gamma}_1^H \hat{\Gamma}_1)\hat{\mathbf{x}} = \hat{D}^H \hat{\mathbf{s}} + \rho \Big(\hat{\mathbf{y}}_2 - \hat{\mathbf{u}}_2 + \hat{\Gamma}_0^H(\hat{\mathbf{y}}_0 - \hat{\mathbf{u}}_0) + \hat{\Gamma}_1^H(\hat{\mathbf{y}}_1 - \hat{\mathbf{u}}_1)\Big) \;. \tag{14}$$

Since $\hat{\Gamma}_0^H \hat{\Gamma}_0$ and $\hat{\Gamma}_1^H \hat{\Gamma}_1$ are diagonal (the $\hat{G}_l$ are diagonal, and therefore so are $\hat{\Gamma}_l$), 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 [17] can be directly applied without any substantial increase in computational cost.

The $\mathbf{y}$ subproblem for (12) can be decomposed into the independent problems

$$\operatorname*{arg\,min}_{\mathbf{y}_2} \lambda \left\|\boldsymbol{\alpha} \odot \mathbf{y}_2\right\|_1 + (\rho/2)\left\|\mathbf{x} - \mathbf{y}_2 + \mathbf{u}_2\right\|_2^2 \tag{15}$$

$$\operatorname*{arg\,min}_{\mathbf{y}_0,\mathbf{y}_1} \mu \left\|\sqrt{\mathbf{y}_0^2 + \mathbf{y}_1^2}\right\|_1 + (\rho/2)\left\|\Gamma_0\mathbf{x} - \mathbf{y}_0 + \mathbf{u}_0\right\|_2^2 + (\rho/2)\left\|\Gamma_1\mathbf{x} - \mathbf{y}_1 + \mathbf{u}_1\right\|_2^2 \;. \tag{16}$$

The solution for (15) is the same as that for (6), and (16) can be solved by use of the block soft thresholding operation [15, Sec. 6.5.1] applied in the same way as in the ADMM algorithm for the standard isotropic TV denoising problem [20, 21], [22, Sec. 4.1], i.e.

$$\mathbf{y}_l = \frac{\mathbf{z}_l}{\sqrt{\mathbf{z}_0^2 + \mathbf{z}_1^2}} \max\left(0, \sqrt{\mathbf{z}_0^2 + \mathbf{z}_1^2} - \frac{\mu}{\rho}\right) \quad l \in \{0, 1\} \tag{17}$$

where $\mathbf{z}_l = \Gamma_l \mathbf{x} + \mathbf{u}_l$ for $l \in \{0, 1\}$.

3.2. Vector TV on Coefficient Maps

Instead of independently applying scalar TV to each coefficient map, one can treat the set of coefficient maps as a multi-channel image and apply Vector TV [19], originally designed for restoration of colour images. The corresponding extension of the CBPDN problem can be written as

$$\operatorname*{arg\,min}_{\{\mathbf{x}_m\}} \frac{1}{2}\left\|\sum_m \mathbf{d}_m * \mathbf{x}_m - \mathbf{s}\right\|_2^2 + \lambda \sum_m \alpha_m \left\|\mathbf{x}_m\right\|_1 + \mu \left\|\sqrt{\sum_m \beta_m \left[(\mathbf{g}_0 * \mathbf{x}_m)^2 + (\mathbf{g}_1 * \mathbf{x}_m)^2\right]}\right\|_1 \;. \tag{18}$$

Using the G_l as defined in Sec. 3.1, the TV term can be written as

$$\mu \left\|\sqrt{\sum_m \beta_m \left[(G_0\mathbf{x}_m)^2 + (G_1\mathbf{x}_m)^2\right]}\right\|_1 \;.$$

Defining $I_B = \begin{pmatrix} I & I & \dots & I \end{pmatrix}$ and

$$\Gamma_l = \begin{pmatrix} \sqrt{\beta_0} G_l & 0 & \dots \\ 0 & \sqrt{\beta_1} G_l & \dots \\ \vdots & \vdots & \ddots \end{pmatrix} \tag{19}$$

allows further reduction to $\mu \left\|\sqrt{I_B(\Gamma_0\mathbf{x})^2 + I_B(\Gamma_1\mathbf{x})^2}\right\|_1$.

Problem (18) can be written in standard ADMM form as

$$\operatorname*{arg\,min}_{\mathbf{x},\mathbf{y}_0,\mathbf{y}_1,\mathbf{y}_2} \frac{1}{2}\left\|D\mathbf{x} - \mathbf{s}\right\|_2^2 + \lambda \left\|\boldsymbol{\alpha} \odot \mathbf{y}_2\right\|_1 + \mu \left\|\sqrt{I_B\mathbf{y}_0^2 + I_B\mathbf{y}_1^2}\right\|_1 \quad \text{s.t.} \begin{pmatrix} \Gamma_0 \mathbf{x} \\ \Gamma_1 \mathbf{x} \\ \mathbf{x} \end{pmatrix} - \begin{pmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = 0 \;. \tag{20}$$

[4]Note that the Γ_l notation is overloaded, taking on a different definition in each section.

The resulting $\mathbf{x}$ subproblem has the same form as (13) and can be solved in the same way. The $\mathbf{y}_2$ subproblem is the same as (15) and can be solved in the same way, while the $\mathbf{y}_0, \mathbf{y}_1$ subproblem, which only differs from (16) in the first term, can be solved by

$$\mathbf{y}_l = \frac{\mathbf{z}_l}{\sqrt{I_B \mathbf{z}_0^2 + I_B \mathbf{z}_1^2}} \max\left(0, \sqrt{I_B \mathbf{z}_0^2 + I_B \mathbf{z}_1^2} - \frac{\mu}{\rho}\right) \quad (21)$$

where $\mathbf{z}_l = \Gamma_l \mathbf{x} + \mathbf{u}_l$ for $l \in \{0, 1\}$.

3.3. Scalar TV in Image Domain

The use of TV regularization here is motivated as an exploration of additional or alternative forms of regularization to the standard ℓ_1 regularization applied to the coefficient maps $\mathbf{x}$. An alternative way of introducing TV regularization, however, would be to consider it as a regularization on the components $D_m \mathbf{x}_m$ of the reconstructed image, which can be written as

$$\begin{aligned}\operatorname*{arg\,min}_{\{\mathbf{x}_m\}} \frac{1}{2}\Big\|\sum_m \mathbf{d}_m * \mathbf{x}_m - \mathbf{s}\Big\|_2^2 + \lambda \sum_m \alpha_m \|\mathbf{x}_m\|_1 + \\ \mu \left\| \sqrt{\Big(\mathbf{g}_0 * \sum_m \beta_m \mathbf{d}_m * \mathbf{x}_m\Big)^2 + \Big(\mathbf{g}_1 * \sum_m \beta_m \mathbf{d}_m * \mathbf{x}_m\Big)^2} \right\|_1 . \quad (22)\end{aligned}$$

The final TV term can be expressed as

$$\mu \left\| \sqrt{\Big(\sum_m \beta_m (\mathbf{g}_0 * \mathbf{d}_m) * \mathbf{x}_m\Big)^2 + \Big(\sum_m \beta_m (\mathbf{g}_1 * \mathbf{d}_m) * \mathbf{x}_m\Big)^2} \right\|_1 .$$

Introducing linear operators $G_{l,m}$ defined such that $G_{l,m}\mathbf{x} = \beta_m(\mathbf{g}_l * \mathbf{d}_m) * \mathbf{x}$, this can be written as

$$\mu \left\| \sqrt{\Big(\sum_m G_{0,m}\mathbf{x}_m\Big)^2 + \Big(\sum_m G_{1,m}\mathbf{x}_m\Big)^2} \right\|_1 ,$$

and defining $\Gamma_l = \begin{pmatrix} G_{l,0} & G_{l,1} & \ldots \end{pmatrix}$ allows further reduction to $\mu \left\| \sqrt{(\Gamma_0 \mathbf{x})^2 + (\Gamma_1 \mathbf{x})^2} \right\|_1$.

Problem (22) can be written in standard ADMM form as

$$\operatorname*{arg\,min}_{\mathbf{x},\mathbf{y}_0,\mathbf{y}_1,\mathbf{y}_2} \frac{1}{2}\|D\mathbf{x} - \mathbf{s}\|_2^2 + \lambda \|\boldsymbol{\alpha} \odot \mathbf{y}_2\|_1 + \mu \left\| \sqrt{\mathbf{y}_0^2 + \mathbf{y}_1^2} \right\|_1$$

$$\text{s.t.} \begin{pmatrix} \Gamma_0 \mathbf{x} \\ \Gamma_1 \mathbf{x} \\ \mathbf{x} \end{pmatrix} - \begin{pmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \\ \mathbf{y}_2 \end{pmatrix} = 0 . \quad (23)$$

The resulting $\mathbf{x}$ subproblem corresponding to (5) has the form

$$\begin{aligned}\operatorname*{arg\,min}_{\mathbf{x}} \frac{1}{2}\|D\mathbf{x} - \mathbf{s}\|_2^2 + \frac{\rho}{2}\|\Gamma_0 \mathbf{x} - \mathbf{y}_0 + \mathbf{u}_0\|_2^2 + \\ \frac{\rho}{2}\|\Gamma_1 \mathbf{x} - \mathbf{y}_1 + \mathbf{u}_1\|_2^2 + \frac{\rho}{2}\|\mathbf{x} - \mathbf{y}_2 + \mathbf{u}_2\|_2^2 . \quad (24)\end{aligned}$$

and the solution of the equivalent DFT domain problem is given by

$$\begin{aligned}(\hat{D}^H \hat{D} + \rho I + \rho \hat{\Gamma}_0^H \hat{\Gamma}_0 + \rho \hat{\Gamma}_1^H \hat{\Gamma}_1)\hat{\mathbf{x}} = \hat{D}^H \hat{\mathbf{s}} + \rho \Big(\hat{\mathbf{y}}_2 - \hat{\mathbf{u}}_2 + \\ \hat{\Gamma}_0^H (\hat{\mathbf{y}}_0 - \hat{\mathbf{u}}_0) + \hat{\Gamma}_1^H (\hat{\mathbf{y}}_1 - \hat{\mathbf{u}}_1)\Big) . \quad (25)\end{aligned}$$

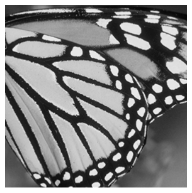
Although the left hand side has the same algebraic form as that of (14), here $\hat{\Gamma}_0^H \hat{\Gamma}_0$ and $\hat{\Gamma}_1^H \hat{\Gamma}_1$ are rank-one rather than diagonal, and can therefore not be grouped together with the ρI term as in the solution for (14). In this case the left hand side is rank-three plus a diagonal: while it cannot be solved using the simple Sherman-Morrison approach, there is still an efficient solution via iterated application of the Sherman-Morrison formula, as used to solve the CBPDN problem for a multi-channel image and dictionary [23]. This involves a greater cost in terms of computation time, but there is a corresponding reduction in memory requirements because $\mathbf{y}_0$ and $\mathbf{y}_1$ are only of the size of the image rather than of the size of the set of coefficient maps.

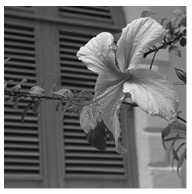
The $\mathbf{y}$ subproblem for (23) has the same form as (15) – (16), and can be solved in the same way.

4. RESULTS

(a) Image 1

(b) Image 2


(c) Image 3


(d) Image 4

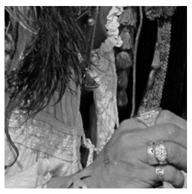

(e) Image 5

Fig. 1. Set of 256×256 pixel noise-free test images.

The performance of standard block-based sparse coding and the different convolutional sparse coding methods described in Sections 2 and 3 was compared on a Gaussian white noise restoration problem. The standard sparse coding was computed via the Basis Pursuit DeNoising (BPDN) problem (i.e. problem (2) where D is a standard dictionary matrix) and the resulting denoised blocks were aggregated via averaging (weighted by the number of blocks covering each pixel) to obtain a denoised image.

Two different dictionaries, one standard and one convolutional, were learned from the same set of ten training images (selected from images on Flickr with a Creative Commons license) of 1024×1024 pixels each. The convolutional dictionary consisted of 128 filters of size 8×8, and was learned via the convolutional dictionary learning algorithm described in [2], while the standard dictionary consisted of 128 vectors of 64 coefficients each (i.e. a vectorised 8×8 image block), and was learned via a non-convolutional variant of the algorithm used for learning the convolutional dictionary, applied to all 8×8 image blocks in the training images. The standard dictionary was used for the BPDN experiments and the convolutional dictionary was used for all CBPDN experiments.

A set of five greyscale reference images, depicted in Fig. 1, was constructed by cropping regions of 256×256 pixels from well-known standard test images. The regions were chosen to contain diversity of content while avoiding large smooth areas, and the size was chosen to be relatively small so that it would be computationally feasible to optimise method parameters via a grid search. The reference images were scaled so that pixel values were in the interval [0, 1], and corresponding test images were constructed by adding Gaussian white noise with a standard deviation of 0.05. Following standard practice [24][6, Sec. 3], the CBPDN decomposition was applied to highpass filtered images, obtained by subtracting a lowpass component computed by Tikhonov regularization [25, pg. 3] with regularization parameter $\lambda_L = 2.0$.

	Test Image				
Method	1	2	3	4	5
BPDN	29.47	32.91	*30.08*	31.73	30.19
CBPDN	29.31	32.70	29.76	31.27	30.09
CBPDN + Grd	29.28	32.76	30.02	31.22	30.12
CBPDN + STV	**30.17**	33.01	29.90	**32.09**	**30.34**
CBPDN + VTV	29.60	**33.04**	**29.96**	31.63	30.31
CBPDN + RTV	29.28	32.84	29.76	31.29	30.19

Table 1. Comparison of denoising performance (PSNR in dB) of the different denoising methods for each of the five test images, with parameters individually optimised for each image. Bold values indicate the best performing CBPDN method. An italic value in the BPDN row indicates that BPDN gave the best overall performance for that image.

For the first set of experiments, the results of which are displayed in Table 1, the denoising performance of the different methods was individually optimised for each image via a search over a logarithmically spaced grid on the λ and μ parameters. The main points worth noting are:

- BPDN is consistently better than CBPDN by a small margin.
- CBPDN + Grd (ℓ_2 of gradient regularization, as in [6, Sec. 4]) gives very similar performance to CBPDN, being slightly better on some test images and slightly worse on others.
- CBPDN + STV (see Sec. 3.1) gives the best overall performance on three of the five test images, with performance within a few tenths of a dB of the best in the other cases. It is consistently better than CBPDN, and better than BPDN in all but one of the test cases.
- In a comparison between CBPDN + STV and CBPDN + VTV (see Sec. 3.2), the former is sometimes better by a moderate margin, but when it is worse this is by a very small amount.
- CBPDN + RTV (see Sec. 3.3) is always worse than the other two TV-augmented CBPDN methods, and is sometimes no better than CBPDN.

The computation times per iteration for the different methods were approximately 0.5 s for BPDN and CBPDN, 0.6 s for CBPDN + Grd, 2.2 s for CBPDN + STV and CBPDN + VTV, and 2.4 s for CBPDN + RTV, i.e. the improved performance of the TV methods is obtained at a significant computational cost.

	Test Image				
Method	1	2	3	4	5
CBPDN + Grd	-2.31	-3.16	-2.51	-1.39	-0.94
CBPDN + STV	+0.04	-0.22	-0.04	-0.03	+0.03
CBPDN + VTV	-0.64	-0.77	-0.89	-0.29	-0.34
CBPDN + RTV	-1.28	-0.66	-0.73	-0.47	-0.33

Table 2. PSNR difference in dB between results for optimisation over both λ and μ (Table 1) and for optimisation over μ only, with $\lambda = 0$.

The second set of experiments evaluated the efficacy of the terms augmenting plain CBPDN by comparing the denoising performance at the best choices of both λ and μ (as in Table 1) with the same method with λ fixed to zero and optimisation only over μ. (There is no need to perform a corresponding comparison with μ fixed to zero since this corresponds to the baseline CBPDN method.) The differences between the PSNR values of the methods optimised over both parameters and only optimised over μ are displayed in Table 2. Note that, for CBPDN + STV, there is a positive difference in two cases and a very small negative difference in two other cases, i.e. for most of the test images, the convolutional representation with only a TV regularization term is competitive with the baseline CBPDN. For all of the other methods the performance is substantially degraded without the ℓ_1 term.

	Test Image				
Method	1	2	3	4	5
BPDN	29.47	32.03	*29.92*	31.38	30.19
CBPDN	29.24	31.73	29.54	30.89	30.00
CBPDN + STV	**29.90**	**32.36**	**29.86**	**31.68**	**30.29**
CBPDN + VTV	29.54	32.35	**29.86**	31.34	30.25
CBPDN + RTV	29.16	32.49	29.76	31.25	30.19

Table 3. Comparison of denoising performance (PSNR in dB) of the different denoising methods for each of the five test images, all with the same parameters obtained by optimising over a separate image set. Bold values indicate the best performing CBPDN method. An italic value in the BPDN row indicates that BPDN gave the best overall performance for that image.

The final set of experiments considers a more realistic scenario in which ground truth is not available for parameter selection for the test images, making it necessary to choose the λ and μ parameters by optimising over a distinct parameter selection image set. The same λ and μ parameters were selected for all test images by finding the values giving the best average performance for a separate image set, again via a search on a logarithmically spaced grid. The results for this experiment are presented in Table 3. Overall, the relative performances of the different methods do not differ qualitatively from those of the experiments reported in Table 1. (CBPDN + Grd is excluded from this set of experiments since it is clear from the first two sets of experiments that it is not competitive.)

5. CONCLUSIONS

While a strictly apples-to-apples comparison between BPDN and CBPDN denoising methods is difficult to construct, the careful attempt reported here indicates that BPDN is slightly superior to baseline CBPDN, but that augmentation of the baseline CBPDN functional with the appropriate TV term substantially boosts performance, surpassing that of BPDN in all but one of the five test cases considered here. With respect to the specific form of additional TV term, scalar TV applied independently to each coefficient map is somewhat superior to a joint vector TV term over all of the coefficient maps, and both of these methods are substantially superior to TV applied in the reconstruction domain rather than to the coefficient maps, indicating that the gain from a TV term on the coefficient maps should not be viewed simply as resulting from denoising via a synthesis of sparse representation and TV image models. It is particularly interesting that the convolutional sparse coding problem with only an STV penalty is competitive in performance with the usual CBPDN form with only an ℓ_1 penalty. At a more abstract level, these results suggest that penalties that exploit the spatial structure of the coefficient maps are necessary to achieve the true potential of the convolutional model.

Implementations of the algorithms proposed here are included in the Python version of the SPORCO library [26, 25].

6. REFERENCES

[1] J. Mairal, F. Bach, and J. Ponce, "Sparse modeling for image and vision processing," *Foundations and Trends in Computer Graphics and Vision*, vol. 8, no. 2-3, pp. 85–283, 2014. doi:10.1561/0600000058

[2] 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

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

[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] Y. Liu, X. Chen, R. K. Ward, and Z. J. Wang, "Image fusion with convolutional sparse representation," *IEEE Signal Process. Lett.*, 2016. doi:10.1109/lsp.2016.2618776

[6] B. Wohlberg, "Convolutional sparse representations as an image model for impulse noise restoration," in *Proc. IEEE Image Video Multidim. Signal Process. Workshop (IVMSP)*, Bordeaux, France, Jul. 2016. doi:10.1109/IVMSPW.2016.7528229

[7] H. Zhang and V. Patel, "Convolutional sparse coding-based image decomposition," in *British Mach. Vis. Conf. (BMVC)*, York, UK, Sep. 2016

[8] T. M. Quan and W.-K. Jeong, "Compressed sensing reconstruction of dynamic contrast enhanced MRI using GPU-accelerated convolutional sparse coding," in *Proc. IEEE Int. Symp. Biomed. Imaging (ISBI)*, Apr. 2016, pp. 518–521. doi:10.1109/ isbi.2016.7493321

[9] H. Zhang and V. M. Patel, "Convolutional sparse and low-rank coding-based rain streak removal," in *Proc. IEEE Winter Conf. Applic. Comput. Vision (WACV)*, March 2017. doi:10.1109/WACV.2017.145

[10] 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

[11] B. Wohlberg, "Convolutional sparse coding with overlapping group norms," Aug. 2017. arXiv:1708.09038

[12] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *Proc. IEEE Global Conf. Signal Info. Process. (GlobalSIP)*, Austin, TX, USA, Dec. 2013, pp. 945–948. doi:10.1109/GlobalSIP.2013.6737048

[13] S. Sreehari, S. V. Venkatakrishnan, B. Wohlberg, G. T. Buzzard, L. F. Drummy, J. P. Simmons, and C. A. Bouman, "Plug-and-play priors for bright field electron tomography and sparse interpolation," *IEEE Trans. Computational Imaging*, vol. 2, no. 4, pp. 408–423, Dec. 2016. doi:10.1109/TCI.2016.2599778

[14] 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

[15] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 127–239, 2014. doi:10.1561/2400000003

[16] 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

[17] 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

[18] L. Rudin, S. J. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms." *Physica D. Nonlin. Phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992. doi:10.1016/0167-2789(92)90242-f

[19] P. Blomgren and T. F. Chan, "Color TV: total variation methods for restoration of vector-valued images," *IEEE Trans. Image Process.*, vol. 7, no. 3, pp. 304–309, March 1998. doi:10.1109/83.661180

[20] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imaging Sci.*, vol. 1, no. 3, pp. 248–272, 2008. doi:10.1137/080724265

[21] J. Yang, W. Yin, Y. Zhang, and Y. Wang, "A fast algorithm for edge-preserving variational multichannel image restoration," *SIAM J. Imaging Sci.*, vol. 2, no. 2, pp. 569–592, 2009. doi:10.1137/080730421

[22] T. Goldstein and S. Osher, "The split Bregman method for l1-regularized problems," *SIAM J. Imaging Sci.*, vol. 2, no. 2, pp. 323–343, 2009. doi:10.1137/080725891

[23] B. Wohlberg, "Convolutional sparse representation of color images," in *Proc. IEEE Southwest Symposium Image Analysis Interpretation (SSIAI)*, Santa Fe, NM, USA, Mar. 2016, pp. 57–60. doi:10.1109/SSIAI.2016.7459174

[24] 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)*, 2011, pp. 2018–2025. doi:10.1109/iccv.2011.6126474

[25] B. Wohlberg, "SPORCO: A Python package for standard and convolutional sparse representations," in *Proceedings of the 15th Python in Science Conference*, Austin, TX, USA, Jul. 2017, pp. 1–8. doi:10.25080/shinma-7f4c6e7-001

[26] ——, "SParse Optimization Research COde (SPORCO)," Software library available from http://purl.org/brendt/software/sporco, 2017