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. While the usual formulation of the convolutional sparse coding problem is slightly inferior to the block-based representations in this problem, 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, Denoising

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
\[ Dx \approx s, \]
where \( D \) is the dictionary, \( x \) is the representation, and \( s \) 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[2] alternative that replace the general linear representation with a sum of convolutions \( \sum_m d_m * x_m \approx s \), where the elements of the dictionary \( d_m \) are linear filters, and the representation consists of the set of coefficient maps \( x_m \), each of which is the same size as \( s \).

The convolutional form has, thus far, received very little attention for image reconstruction problems, but there are indications that interest is growing, with recent work addressing applications in superresolution [3], image fusion [4], impulse noise denoising [5], structure/texture decomposition [6], and dynamic MRI reconstruction [7]. Surprisingly, denoising of Gaussian white noise, arguably the simplest of all imaging inverse problems, has received no attention beyond a very brief example providing insufficient detail for reproducibility [8 Sec. 4.4]. The present paper argues that, despite its numerous advantages in many contexts, the convolutional form is not very competitive for the Gaussian white noise denoising problem, but that these deficiencies can be mitigated by moving beyond simple \( \ell_1 \) regularization, the specific form being investigated here consisting of additional penalties on the gradients of the coefficient maps. The only other work that has considered such a modification introduced a term based on the graph Laplacian of the non-local image graph, which was found to be effective, but computationally expensive [9].

2. CONVOLUTIONAL SPARSE CODING

The most widely used form of convolutional sparse coding is Convolutional Basis Pursuit DeNoising (CBPDN), defined as
\[
\arg\min_{x_m} \frac{1}{2} \sum_m d_m * x_m - s \| s \|_2 + \lambda \sum_m \alpha_m \| x_m \|_1,
\]
where the \( \alpha_m \) allow distinct weighting of the \( \ell_1 \) term for each filter \( d_m \). At present, the most efficient approach to solving this problem [2] is via the Alternating Direction Method of Multipliers (ADMM) [10] framework. An outline of this method is presented here as a basis for extensions proposed in following sections.

This problem can be expressed in ADMM standard form as
\[
\arg\min_{x,y} \frac{1}{2} \| D x - s \|_2^2 + \lambda \| \alpha \circ y \|_1 \quad \text{s.t.} \quad x - y = 0,
\]
which can be solved via the ADMM iterations
\[
x^{(j+1)} = \arg\min_x \frac{1}{2} \| D x - s \|_2^2 + \frac{\rho}{2} \| x - y^{(j)} + u^{(j)} \|_2^2
\]
\[
y^{(j+1)} = \arg\min_y \lambda \| \alpha \circ y \|_1 + \frac{\rho}{2} \| x^{(j+1)} - y + u^{(j)} \|_2^2
\]
\[
u^{(j+1)} = u^{(j)} - x^{(j+1)} - y^{(j+1)}.
\]
The only computationally expensive step is [5], which can be solved via the equivalent DFT domain problem
\[
\arg\min_{\hat{x}} \frac{1}{2} \| \hat{D} \hat{x} - \hat{s} \|_2^2 + \frac{\rho}{2} \| \hat{x} - (\hat{y} - \hat{u}) \|_2^2,
\]
where \( \hat{D} = (\hat{D}_0 \hat{D}_1 \ldots) \) and
\[
\hat{x} = \begin{pmatrix} \hat{x}_0 \\ \hat{x}_1 \\ \vdots \end{pmatrix}, \quad \hat{y} = \begin{pmatrix} \hat{y}_0 \\ \hat{y}_1 \\ \vdots \end{pmatrix}, \quad \hat{u} = \begin{pmatrix} \hat{u}_0 \\ \hat{u}_1 \\ \vdots \end{pmatrix},
\]
with \( z \) denoting the DFT of variable \( z \). The solution for (8) is given by the \( MN \times MN \) linear system (for \( M \) filters and an image \( s \) with \( N \) pixels)

\[
(\hat{D}^H \hat{D} + \rho I)\hat{x} = \hat{D}^H \hat{s} + \rho (\hat{y} - \hat{u}).
\] (10)

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 [11], each of which has a system matrix consisting of the sum of rank-one and diagonal terms so they can be solved very efficiently by exploiting the Sherman-Morrison formula [12].

3. GRADIENT REGULARIZATION

An extension of (1) to include \( \ell_2 \) regularization on the gradients of the coefficient maps was proposed in [5]. 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 [5]. Considering the edge-smoothing effect of \( \ell_1 \) gradient regularization, a reasonable alternative to consider is Total Variation (TV) regularization, i.e. the \( \ell_1 \) norm of the gradient. We consider three different variants:

1. scalar TV [13] applied independently to each coefficient map,
2. vector TV [4] applied jointly to the set of coefficient maps, and
3. scalar TV [13] applied to the reconstructed image rather than to the coefficient maps.

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

\[
\arg\min_{x_m} \frac{1}{2} \left\| \sum_m d_m * x_m - s \right\|^2_2 + \lambda \sum_m \alpha_m \left\| x_m \right\|_1 + \mu \sum_m \beta_m \left\| (g_0 * x_m)^2 + (g_1 * x_m)^2 \right\|_1,
\] (11)

where \( g_0 \) and \( g_1 \) are filters that compute the gradients along image rows and columns respectively. The TV term can be written as \( \mu \sum_m \beta_m \left\| \sqrt{(G_0 x_m)^2 + (G_1 x_m)^2} \right\|_1 \), where linear operators \( G_0 \) and \( G_1 \) are defined such that \( G_i x_m = g_i * x_m \), and defining

\[
\Gamma_i = \begin{pmatrix} \beta_0 G_i & 0 & \ldots \\ 0 & \beta_1 G_i & \ldots \\ \vdots & \vdots & \ddots \end{pmatrix}
\] (12)

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

Problem (11) can be written in standard ADMM form as

\[
\arg\min_{x,y_0,y_1,y_2} \frac{1}{2} \left\| D x - s \right\|^2_2 + \lambda \left\| \alpha \odot y_2 \right\|_1 + \mu \left\| \sqrt{y_0^2 + y_1^2} \right\|_1
\]

s.t. \( \begin{pmatrix} \Gamma_0 x \\ \Gamma_1 x \\ x \end{pmatrix} - \begin{pmatrix} y_0 \\ y_1 \\ y_2 \end{pmatrix} = 0 \). (13)

The resulting \( x \) subproblem corresponding to (5) has the form

\[
\arg\min_x \frac{1}{2} \left\| D x - s \right\|^2_2 + \frac{\rho}{2} \left\| \Gamma_0 x - y_0 + u_0 \right\|^2_2 + \frac{\rho}{2} \left\| \Gamma_1 x - y_1 + u_1 \right\|^2_2 + \frac{\rho}{2} \left\| x - y_2 + u_2 \right\|^2_2,
\] (14)

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

\[
(\hat{D}^H \hat{D} + \rho I) \hat{\Gamma}_0 \hat{x} + \rho \hat{\Gamma}_1 \hat{x} = \hat{D}^H \hat{\Gamma}_0 \hat{s} + \rho (\hat{y}_2 - \hat{u}_2 + \hat{\Gamma}_0 \hat{u}_0 + \hat{\Gamma}_1 \hat{u}_1) \hat{u}_2.
\] (15)

Since \( \hat{\Gamma}_0 \hat{\Gamma}_0 \hat{\Gamma}_1 \) are diagonal they can be grouped together with the \( \rho I \) term; the independent linear systems described in Sec. 3 can be directly applied without any substantial increase in computational cost.

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 [14], originally designed for restoration of colour images. The corresponding extension of the CBPDN problem can be written as

\[
\arg\min_{x_m} \frac{1}{2} \left\| \sum_m d_m * x_m - s \right\|^2_2 + \lambda \sum_m \alpha_m \left\| x_m \right\|_1 + \mu \left\| \sum_m \beta_m \left( (g_0 * x_m)^2 + (g_1 * x_m)^2 \right) \right\|_1.
\] (16)

The TV term can be written as

\[
\mu \left\| \sum_m \beta_m \left( (G_0 x_m)^2 + (G_1 x_m)^2 \right) \right\|_1.
\]

where the \( G_i \) are as defined in Sec. 3.1 and defining

\[
\Gamma_i = \begin{pmatrix} \sqrt{\beta_0 G_i} & 0 & \ldots \\ 0 & \sqrt{\beta_1 G_i} & \ldots \\ \vdots & \vdots & \ddots \end{pmatrix}
\] (17)

and \( I_B = (I \ I \ \ldots \ I) \) allows further reduction to

\[
\mu \left\| I_B \Gamma_0 x) x + I_B \Gamma_1 x \right\|_1.
\]

Problem (16) can be written in standard ADMM form as

\[
\arg\min_{x,y_0,y_1,y_2} \frac{1}{2} \left\| D x - s \right\|^2_2 + \lambda \left\| \alpha \odot y_2 \right\|_1 + \mu \left\| I_B y_0 + I_B y_1 + I_B y_2 \right\|_1
\]

s.t. \( \begin{pmatrix} \Gamma_0 x \\ \Gamma_1 x \\ x \end{pmatrix} - \begin{pmatrix} y_0 \\ y_1 \\ y_2 \end{pmatrix} = 0 \). (18)

The resulting \( x \) subproblem corresponding to (5) is the same form as (14), and can be solved in the same way.

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 \( \ell_1 \) regularization applied to the coefficient maps \( x \). An alternative way of introducing TV regularization, however, would be to consider it as
The resulting step applied to the result, as in \[6\].

Solving the CBPDN problem followed by an additional TV denoising step applied to the result, as in \[6\].

Note that the solution to this problem could be approximated by \(\hat{\Gamma} = \Gamma_0\) and defining \(\Gamma = (\hat{\Gamma}_0, \hat{\Gamma}_1, \ldots)\) allows further reduction to \(\mu \sqrt{\hat{\Gamma}_0 x^2 + (\hat{\Gamma}_1 x)^2} = 1\).

Problem \((19)\) can be written in standard ADMM form as

\[
\begin{align*}
\arg\min_{x, y_0, y_1, y_2} & \frac{1}{2} \|Dx - s\|^2 + \lambda \|\alpha \odot y_2\|_1 + \mu \sqrt{y_0^2 + y_1^2} \\
\text{s.t.} \quad & \begin{pmatrix} \hat{\Gamma}_0 x \\ \hat{\Gamma}_1 x \end{pmatrix} - \begin{pmatrix} y_0 \\ y_1 \\ y_2 \end{pmatrix} = 0.
\end{align*}
\]

The resulting \(x\) subproblem corresponding to \((5)\) has the form

\[
\begin{align*}
\arg\min_x & \frac{1}{2} \|Dx - s\|^2 + \frac{\rho}{2} \|\hat{\Gamma}_0 x - y_0 + u_0\|^2 + \\
& + \frac{\rho}{2} \|\hat{\Gamma}_1 x - y_1 + u_1\|^2 + \frac{\rho}{2} \|x - y_2 + u_2\|^2.
\end{align*}
\]

The diversity of content while avoiding large smooth areas, and the size of the image rather than the size of the set of coefficient maps.

4. RESULTS

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 \(\times\) 1024 pixels each. The convolutional dictionary consisted of 128 filters of size 8 \(\times\) 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 \(\times\) 8 image block), and was learned via a non-convolutional variant of the algorithm used for learning the convolutional dictionary, applied to all 8 \(\times\) 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 \(\times\) 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.

| Method       | Test Image | 1    | 2    | 3    | 4    | 5    |
|--------------|------------|------|------|------|------|------|
| BPDN         |            | 29.47| 32.91| 30.08| 31.73| 30.19|
| CBPDN        |            | 29.31| 32.70| 32.76| 31.27| 30.09|
| CBPDN + Grd  |            | 29.28| 32.76| 30.02| 31.22| 30.12|
| CBPDN + STV  |            | 30.17| 33.01| 30.90| 32.09| 30.34|
| CBPDN + VTV  |            | 29.60| **33.04**| 29.96| 31.63| 30.31|
| CBPDN + RTV  |            | 29.28| 32.84| 31.29| 31.09| 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 method independently for the BPDN method and the CBPDN methods; a bold value in the BPDN row indicates that BPDN gave the best performance for that image, while a bold value in a CBPDN row indicates the best of the CBPDN methods, which is also the overall best method if the corresponding BPDN value is not bold.

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 \(\lambda\) and \(\mu\) parameters. The main points worth noting are:

- **BPDN** is consistently better than CBPDN by a small margin.
- **CBPDN + Grd** (i.e. \(\ell_2\) of gradient regularisation) 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, 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. 3.3), 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.

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

| Method        | Test Image |
|---------------|------------|
| CBPDN + Grd   | -2.31      |
| CBPDN + STV   | +0.04      |
| CBPDN + VTV   | -0.64      |
| CBPDN + RTV   | -1.28      |

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 method independently for the BPDN method and the CBPDN methods: a bold value in the BPDN row indicates that BPDN gave the best performance for that image, while a bold value in a CBPDN row indicates the best of the CBPDN methods, which is also the overall best method if the corresponding BPDN value is not bold.

| Method        | Test Image |
|---------------|------------|
| BPDN          | 29.47      |
| CBPDN         | 29.24      |
| CBPDN + STV   | **32.36**  |
| CBPDN + VTV   | 29.54      |
| CBPDN + RTV   | 29.16      |

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.

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. Since there is no reason to expect that the standard TV methods presented here are optimal in this role, other existing methods, such as alternative forms of TV (e.g. [16][17]), deserve consideration, and the development of new penalties designed specifically for this context requires attention.

Implementations of the algorithms proposed here will be included in a future release of the SPORCO library [18].
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:10.1561/06000000058

[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] 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

[4] 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

[5] 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

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

[7] 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

[8] 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

[9] X. Luo and B. Wohlberg, “Convolutional Laplacian sparse coding,” in Proc. IEEE Southwest Symposium Image Analysis Interpretation (SSIAI), Santa Fe, NM, USA, Mar. 2016. pp. 133–136. doi:10.1109/SSIAI.2016.7459193

[10] 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:10.1561/12000000016

[11] 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

[12] 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

[13] 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)90242-4

[14] 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

[15] 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

[16] S. Lefkimmiatis, J. P. Ward, and M. Unser, “Hessian Schatten-norm regularization for linear inverse problems,” IEEE Trans. Image Process., vol. 22, no. 5, pp. 1873–1888, May 2013. doi:10.1109/TIP.2013.2237919

[17] S. Lefkimmiatis, A. Roussos, P. Maragos, and M. Unser, “Structure tensor total variation,” SIAM J. Imaging Sci., vol. 8, no. 2, pp. 1090–1122, 2015. doi:10.1137/14098154X

[18] B. Wohlberg, “SParse Optimization Research COde (SPORCO),” Software library available from [http://purl.org/brendt/software/sporco], 2016.