FAST CONVOLUTIONAL SPARSE CODING WITH SEPARABLE FILTERS

Gustavo Silva, Jorge Quesada, Paul Rodríguez

Department of Electrical Engineering
Pontificia Universidad Católica del Perú
Lima, Peru

Brendt Wohlberg

T-5 Applied Mathematics and Plasma Physics
Los Alamos National Laboratory
Los Alamos, NM 87545, USA

ABSTRACT

Convolutional sparse representations (CSR) of images are receiving increasing attention as an alternative to the usual independent patch-wise application of standard sparse representations. For CSR the dictionary is a filter bank of non-separable 2D filters, and the representation itself can be viewed as the synthesis dual of the analysis representation provided by a single level of a convolutional neural network (CNN). The current state-of-the-art convolutional sparse coding (CSC) algorithms achieve their computational efficiency by applying the convolutions in the frequency domain.

It has been shown that any given 2D non-separable filter bank can be approximated as a linear combination of a relatively small number of separable filters. This approximation has been exploited for computationally efficient CNN implementations, but has thus far not been considered for convolutional sparse coding. In this paper we propose a computationally efficient algorithm, that apply the convolution in the spatial domain, to solve the CSC problem when the corresponding dictionary filters are separable. Our algorithm, based on the ISTA framework, use a two-term penalty function to attain competitive results when compared to the state-of-the-art methods in terms of computational performance, sparsity and reconstruction quality.

Index Terms— Convolutional Sparse Representation, Convolutional Sparse Coding, Separable Filters

1. INTRODUCTION

Convolutional sparse representation (CSR) [1, 2] models an entire signal or image as a sum over a set of convolutions of coefficient maps, of the same size as the signal or image, with their corresponding dictionary filters. Given a set of dictionary filters, which in general are non-separable, the most widely used formulation of the convolutional sparse coding (CSC) problem is Convolutional BPDN (CBPDN) [3], defined as

$$\underset{\{\mathbf{u}_k\}}{\arg\min} \;\; \frac{1}{2} \left\| \sum_{k=1}^{K} H_k * \mathbf{u}_k - \mathbf{b} \right\|_2^2 + \lambda \sum_{k=1}^{K} p(\mathbf{u}_k) \tag{1}$$

where $\{H_k\}$ is a set of K non-separable $L_1 \times L_2$ filters, $\{\mathbf{u}_k\}$ is the corresponding set coefficient maps (each with $N = N_1 \times N_2$ samples), $\mathbf{b}$ is the $N_1 \times N_2$ input image, and the penalty function $p(\mathbf{x}) = \|\mathbf{x}\|_1$.

It has been shown [4, 5, 6] that any 2D filter bank (FB), composed of a large number of non-separable filters $\{H_k\}$, can be approximated as a linear combination of a relatively small number of separable filters $\{G_r\}$, i.e.

$$H_k \approx \sum_{r=1}^{R} \alpha_{kr} G_r \quad k \in \{1, 2, \ldots, K\}, \tag{2}$$

where it is desirable that $R \ll K$. Assuming that (2) is an equality, via simple algebraic manipulation we have

$$\sum_{k=1}^{K} H_k * \mathbf{u}_k = \sum_{r=1}^{R} G_r * \left(\sum_{k=1}^{K} \alpha_{kr} \mathbf{u}_k \right) \; . \tag{3}$$

The spatial domain computation of the left- and right-hand sides of (3) require $\mathcal{O}(2 \cdot K \cdot (N_1 \cdot N_2) \cdot (L_1 \cdot L2))$ and $\mathcal{O}(4 \cdot R \cdot (N_1 \cdot N_2) \cdot (L_1 + L_2) \; + \; 2 \cdot K \cdot R \cdot (N_1 \cdot N_2))$ operations respectively.

In this paper we present a FISTA/NIHT based algorithm to solve

$$\underset{\{\mathbf{u}_k\}}{\arg\min} \;\; \frac{1}{2} \left\| \sum_{r=1}^{R} G_r * \left(\sum_{k=1}^{K} \alpha_{kr} \mathbf{u}_k \right) - \mathbf{b} \right\|_2^2 + \lambda \sum_{k=1}^{K} p(\mathbf{u}_k) \; , \tag{4}$$

where $p(\cdot)$ is a two-term penalty function[1] given by

$$p(\mathbf{x}) = \alpha \|\mathbf{x}\|_1 + \beta \phi_{\mathrm{nng}}(\mathbf{x}) \; , \tag{5}$$

with $\alpha > 0$, $\beta > 0$, and $\phi_{\mathrm{nng}}(\mathbf{x})$ is the penalty function associated with the Non-Negative Garrote (NNG) thresholding rule [8]. While the specific choice of (5) has not been reported before, it is related to [9, 10, 11, 12], where different penalty functions, which induce sparsity more strongly than the ℓ_1-norm penalty function, have been studied. We will further motivate (5) in Section 3.1.

The proposed algorithm is also used to solve

$$\underset{\{\mathbf{v}_k\}}{\arg\min} \;\; \frac{1}{2} \left\| \sum_{r=1}^{R} G_r * \mathbf{v}_r - \mathbf{b} \right\|_2^2 + \lambda \sum_{r=1}^{K} p(\mathbf{v}_r) \; , \tag{6}$$

with $p(\cdot)$ defined as in (5), and where the $\{\mathbf{v}_r\}$ can be interpreted as "separable coefficient maps", and the $\{\mathbf{u}_k\}$ are the corresponding non-separable coefficient maps, related by

$$\mathbf{v}_r = \sum_{k=1}^{K} \alpha_{kr} \mathbf{u}_k \; . \tag{7}$$

The advantage of problem (6) is that it is cheaper to solve than (4) in terms of both computation time and memory requirements.

Our computational results, presented in Section 4, show that the proposed algorithm, which applies the convolution operation in the spatial domain, when solving either (4) or (6), are competitive with the state-of-the-art algorithms [13, 3] for which the convolution operation is computed in the Fourier domain. Furthermore, the com-

[1] It is shown in [7] that (4) is convex when the penalty function $p(\cdot)$ is defined as in (5).

putational performance of solving (6) can be up to five times faster than the state-of-the-art algorithms, while obtaining competitive reconstruction quality and sparsity.

2. PREVIOUS RELATED WORK

We start this section with a brief review of the well-known ISTA framework to then provide a list of computational algorithms that target the solution of (1).

2.1. ISTA and Variants

For either (1) or (4) it is possible to find the matrix H such that

$$H\mathbf{u} = \sum_{k=1}^{K} H_k * \mathbf{u}_k = \sum_{r=1}^{R} G_r * \mathbf{v}_r \ , \tag{8}$$

where $\mathbf{v}_r$ is defined in (7) and $\mathbf{u} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K]$, so (1) or (4) can be written as

$$\underset{\{\mathbf{u}\}}{\arg\min} \ \ f(\mathbf{u}) + \lambda \cdot p(\mathbf{u}) \ . \tag{9}$$

For (9), $\nabla f(\mathbf{u}) = H^T(H\mathbf{u} - \mathbf{b})$. When H^T is approximated by the separable filters then

$$H^T\mathbf{x} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_R] \cdot \boldsymbol{\alpha}, \tag{10}$$

where $\mathbf{z}_r = G_r \circ \mathbf{x}$, $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \mathbf{u}_K]^T$, $\boldsymbol{\alpha}_r = [\alpha_{1r}, \alpha_{2r}, \ldots, \alpha_{Kr}]^T$, and $\circ$ represents correlation[2]. It is worth noting that spatial computation of $H^T\mathbf{x}$, when using the separable filters $\{G_r\}$, has the same cost as $H\mathbf{x}$ (see (3)).

When $p(\mathbf{u}) = \|\mathbf{u}\|_1$, the corresponding thresholding rule in ISTA is soft thresholding i.e. $\mathrm{shrink}(x, \lambda) = \mathrm{sign}(x)\max\{0, |x| - \lambda\}$, while for $p(\mathbf{u}) = \|\mathbf{u}\|_0$ it is hard thresholding, i.e. $\mathrm{hard}(x, \lambda) = \mathrm{I}_{[|x|>\lambda]} \cdot x$, where $\mathrm{I}_{[\mathrm{cond}]}$ is the indicator function. Both of these choices have drawbacks such as discontinuity, estimation bias, etc. [14, Section 3].

Recently, [9, 10, 11, 12] have explored the use of alternatives to the usual ℓ_1-norm penalty function and thoroughly assessed the convergence properties of the ISTA/FISTA algorithm for any thresholding rule[3]. Interestingly, ISTA using a penalty function other than the ℓ_1-norm can obtain better reconstruction quality for problems that promote sparsity than hard/soft thresholding [9, 10, 11, 12].

2.1.1. ISTA

The Iterative Shrinkage/Thresholding algorithm (ISTA) is a well-known first-order solver for (9) that achieves a sublinear rate of convergence. For convenience the computational steps of ISTA are reproduced in Algorithm 1.

Inputs: λ (parameter), $\mathbf{u}^0$ (initial guess)
Step n: $(n \geq 1)$ Compute
1 $\mu_n \in [0, \ \frac{1}{\|H^T H\|}]$;
2 $\mathbf{u}^n = \mathrm{thresh}(\mathbf{u}^{n-1} + \mu_n H^T(\mathbf{b} - H\mathbf{x}^{n-1}), \ t_n\lambda)$

Algorithm 1: ISTA applied to (9) when $f(\mathbf{u}) = \frac{1}{2}\|H\mathbf{u} - \mathbf{b}\|_2^2$. $\mathrm{thresh}(\cdot)$, in line 2, is the corresponding thresholding rule[5] for $p(\mathbf{u})$.

Note that usually $\mu_n = \frac{1}{\tau}$, where $\tau = 2 \cdot \sigma_{\max}(H^T H)$ is the smallest Lipschitz constant of $\nabla f(\mathbf{u})$, although it can also be updated for each iteration [15, Section 3].

2.1.2. FISTA

The Fast Iterative Shrinkage Thresholding algorithm (FISTA) [15] is closely related to ISTA, but has faster convergence. For convenience the computational steps of FISTA are reproduced in Algorithm 2. Note that L in this algorithm can be updated for each iteration by following a backtracking step size rule [15, Section 4].

Inputs: λ (parameter), L (Lipschitz constant of $\nabla f(\mathbf{u})$)
Step 0: Set $\mathbf{y}_1 = \mathbf{u}_0$ (initial guess), $\beta_1 = 1$
Step n: $(n \geq 1)$ Compute
1 $\mathbf{u}^n = \mathrm{thresh}(\mathbf{u}^{n-1} + \frac{1}{L}H^T(\mathbf{b} - H\mathbf{y}^n), \frac{\lambda}{L})$
2 $\beta_{n+1} = \frac{1+\sqrt{1+4\beta_n}}{2}$
3 $\mathbf{y}^{n+1} = \mathbf{u}^n + \frac{\beta_n - 1}{\beta_{n+1}}(\mathbf{u}^n - \mathbf{u}^{n-1})$

Algorithm 2: FISTA applied to (9) when $f(\mathbf{u}) = \frac{1}{2}\|H\mathbf{u} - \mathbf{b}\|_2^2$. $\mathrm{thresh}(\cdot)$, in line 1, is the corresponding thresholding rule[5] for $p(\mathbf{u})$.

2.1.3. Normalized IHT

A simple modification to the Iterative Hard Thresholding (IHT) was proposed in [16], along with an assessment of the performance and stability of the resulting algorithm, called Normalized IHT (NIHT). IHT / NHIT can also be used to solve (9) with $p(\mathbf{u}) = \|\mathbf{u}\|_0$ (i.e., the ℓ_0 "norm" counting the number of non-zero elements in $\mathbf{u}$). The NIHT is reproduced in Algorithm 3 for convenience.

Inputs: λ
Step 0: $\mathbf{u}^0$: initial guess
Step n: $(n \geq 1)$ Compute
1 $\boldsymbol{\nu}^n = H^T(\mathbf{b} - H\mathbf{u}^n), \qquad \mu_n = \frac{(\boldsymbol{\nu}^n)^T\boldsymbol{\nu}^n}{(\boldsymbol{\nu}^n)^T H^T H \boldsymbol{\nu}^n}$
2 $\mathbf{u}^n = \mathrm{hard}(\mathbf{u}^{n-1} + \mu_n\boldsymbol{\nu}^n, \mu_n\lambda)$

Algorithm 3: IHT applied to (9) when $f(\mathbf{u}) = \frac{1}{2}\|H\mathbf{u} - \mathbf{b}\|_2^2$ and $p(\mathbf{u}) = \|\mathbf{u}\|_0$.

2.2. Methods for CSC

Several algorithms have been proposed to directly solve the original CSC problem (1). Earlier approaches were based on ISTA [17] or FISTA [18], computing the necessary gradient in the spatial domain, assuming a non-separable filter bank $\{H_k\}$. More recent approaches are based on the ADMM framework, computing the convolutions in the frequency domain [19, 13, 20, 3, 21].

3. PROPOSED METHOD

The proposed FISTA/NIHT-based algorithm is described in this section. Full details of the novel aspects are provided in Section 3.1, and the shortcomings of ADMM-based algorithms for separable filter banks are outlined in Section 3.2.

[2] Equivalent to performing convolution with a kernel rotated 180 degrees.

[3] Given a thresholding rule, a corresponding penalty function $p(\cdot)$ can always be computed [14, 10, 11].

[5] If $p(\mathbf{u}) = \|\mathbf{u}\|_1$, then the corresponding thresholding rule is soft thresholding, i.e. $\mathrm{thresh}(x, \lambda) = \mathrm{shrink}(x, \lambda)$. For alternatives see [9, 10, 11, 12].

3.1. Hybrid FISTA/NIHT based

The proposed algorithm, listed in Algorithm 4, includes two particular characteristics: (i) it incorporates the optimal step size rule of NIHT [16, eq. (18)] in the FISTA algorithm, and (ii) it uses the two-term penalty function described in (5). When the CSC dictionary is separable, the NIHT optimal step size rule (μ_n in line 2, Algorithm 4) can be efficiently computed: the numerator and denominator of μ_n are given by inner products with the same size as the input image, while the computation of these vectors is a by product of the FISTA procedure, carried out via (8) and (10).

Inputs: λ
Step 0: Set $\mathbf{y}^{(1)} = \mathbf{z}^{(0)}$ (initial guess for (4) or (6)), $\beta_1 = 1$
Step n: ($n \geq 1$) Compute
1 c_n : compute ℓ_1-cost for $\mathbf{z}^{n-1}$.
2 $\boldsymbol{\xi}^{(n)} = H^T(\mathbf{b} - H\mathbf{y}^{(n)}), \quad \boldsymbol{\nu} = H\boldsymbol{\xi}^{(n)}, \qquad \mu_n = \frac{(\boldsymbol{\xi}^{(n)})^T\boldsymbol{\xi}^{(n)}}{\boldsymbol{\nu}^T\boldsymbol{\nu}}$
3 $\mathbf{z}^{(n)} = \text{thresh}(\mathbf{z}^{(n-1)} + \frac{1}{\mu_n}\boldsymbol{\xi}^{(n)}, \frac{\lambda}{\mu_n}, \alpha, \beta)$
4 $\beta_{n+1} = \frac{1+\sqrt{1+4\beta_n}}{2}, \qquad \mathbf{y}^{(n+1)} = \mathbf{z}^{(n)} + \frac{\beta_n - 1}{\beta_{n+1}}(\mathbf{z}^{(n)} - \mathbf{z}^{(n-1)})$

Algorithm 4: Proposed FISTA/NIHT-based algorithm to solve (4) or (6). $H\mathbf{x}$ and $H^T\mathbf{x}$ are computed via (3) and (10) respectively for the former, whereas for the latter those same equations apply considering $\alpha_{rr} = 1$ and zero otherwise. $\text{thresh}(\cdot)$, in line 3, is the corresponding thresholding rule[6].

When the ℓ_1-norm penalty function is considered for either (4) or (6), we noticed[7] that while the sparsity of the solution obtained via Algorithm 4 was competitive, the reconstruction error underperformed that obtained when solving (1) via any of the state-of-the-art algorithms listed in Section 2.2. When the ℓ_0-norm or the penalty function associated with the NNG thresholding rule were used[7], the reconstruction error greatly improved at the cost of the undesirable effect of also increasing the sparsity measure by a factor of two or more. Furthermore, when the MSE / Sparsity curve is plotted for the above considered cases, as it is done for our final results depicted in Figure 1.b or 1.c, the curve for the ℓ_1-norm penalty function resembles an up-shifted version (i.e. competitive sparsity, relatively poor MSE) of that of [3] (i.e. state-of-the-art, red line in Figure 1.b or 1.c), whereas the curves for the ℓ_0-norm or the penalty function associated with the NNG thresholding rule resemble a left-shifted version (relatively poor sparsity, competitive MSE) of that obtained by [3].

Considering the results presented in [9, 10, 11, 12], we also conducted experiments by setting the penalty cost function $p(\cdot)$, for either (4) or (6), to other non ℓ_1-norm alternatives. We observed (see [7] for details) that the best performance trade-off, which balances the reconstruction error, sparsity measure and time performance is obtained when the penalty function described in (5) is used, whose thresholding rule is given by

$$\text{thresh}(x, \lambda, \alpha, \beta) = \begin{cases} \gamma(x, \lambda, \alpha, \beta) & \text{if } |x| > \lambda \cdot (\alpha + \beta) \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

where $\gamma(x, \lambda, \alpha, \beta) = \frac{x\left(x - \text{sign}(x)\frac{2}{\lambda \cdot \alpha}\right) + \lambda^2 \cdot \alpha^2 - \lambda^2 \cdot \beta^2}{x + \lambda \cdot \alpha}$.

When applying the constraint $\alpha + \beta = 1$, thresholding rule (11) is in fact a synthesis between soft-thresholding and NNG, parameterized by the constants α, β. In order to balance the effects (observed in our preliminary experimental results[7]) of using the ℓ_1-norm or the penalty function associated with the NNG thresholding rule, we propose to vary, at each iteration of Algorithm 4, the values of α and β, starting with α close to 1, and then slowly decreasing it. The computational results in Section 4 indicate that this simple strategy is very effective.

The particular choice of (5) as the penalty function raises a question about the convergence of Algorithm 4 to the global minimizer of (4) or (6). A detailed analysis on this regard is given in [7], but we note here that it is shown in [12] that (4) or (6) are convex as long as the penalty function $p(\cdot)$ is ρ-weakly convex[8]; in [12] it is also shown that for the aforementioned case, an ISTA based algorithm is guaranteed to converge to the global minimizer. It is straightforward to prove that the penalty function defined in (5) is indeed ρ-weakly convex with $\rho = \frac{1}{2}$.

3.2. Shortcomings of an ADMM-based algorithm for (4)

The ADMM algorithm [22] is a well-known and versatile method for solving optimization problems of the form $\min_{\mathbf{u},\mathbf{v}} f(\mathbf{u}) + \lambda \cdot p(\mathbf{v}) \quad \text{s.t.} \quad A\mathbf{u} + B\mathbf{v} - \mathbf{c} = 0$. Problem (4) can be expressed in this form as

$$\min_{\{\mathbf{u}_k, \mathbf{y}_k\}} \frac{1}{2}\|H\mathbf{u} - \mathbf{b}\|_2^2 + \lambda \sum_{k=1}^{K} p(\mathbf{y}_r) \quad \text{s.t.} \quad \mathbf{y}_k = \mathbf{u}_k \;, \tag{12}$$

where $\mathbf{u} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K]$ and $H\mathbf{u} = \sum_{r=1}^{R} G_r * \Big(\sum_{k=1}^{K} \alpha_{kr}\mathbf{u}_k\Big)$. The ADMM steps for (12) are exactly those considered in [3, see equations (10)–(12)]. Of these, the only computationally expensive step is

$$\min_{\mathbf{u}} \; \frac{1}{2}\|H\mathbf{u} - \mathbf{b}\|_2^2 + \frac{\rho}{2}\sum_{k=1}^{K}\|\mathbf{u}_k - \mathbf{y}_k^{(j)} + \mathbf{z}_k^{(j)}\|_2^2 \;. \tag{13}$$

Solving this sub-problem in the Fourier domain leads to the algorithm proposed in [3]. Efficient solution in the spatial domain is far more difficult, the most effective option being conjugate gradient (CG). Since each CG solve requires multiple computations of $H\mathbf{x}$ and $H^T\mathbf{x}$, the computational performance is greatly inferior to that of the Fourier domain approach.

4. RESULTS

All of the experiments reported here use the dictionary filter bank $\{H_k\}$ consisting of 144 filters of size 12×12 distributed with the SPORCO library [23]. Images from the USC-SIPI database [24] were used as test images for CSC problem (1), (4) and (6).

All CSC related experiments were carried on an Intel i7-4710HQ (2.5 GHz, 6MB Cache, 32GB RAM) based laptop with a nvidia GTX980M GPU card. The GPU-enabled Matlab SPORCO library [23] was used to solve (1), since it is the state-of-the-art for this problem. To solve either (4) or (6) we developed a GPU-enabled Matlab library, which has been made publicly available [25].

We consider two separable filter banks with 24 and 30 separable filters[9] to approximate the original filter bank $\{H_k\}$. These separable filter banks were computed via [26] (other alternatives include [27, 4, 5], but computational examples [26, 25] indicate that the results are indistinguishable). The parameters α, β, of the thresholding

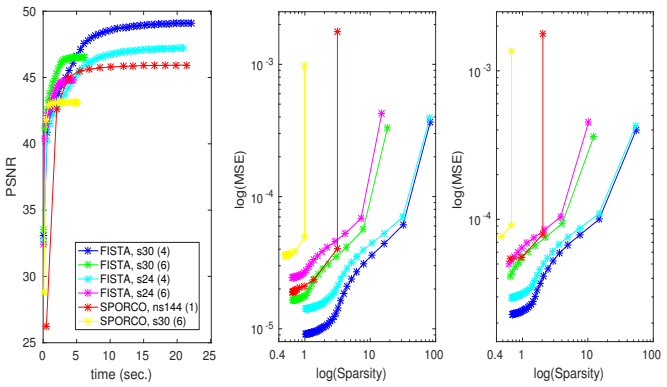
[6] If the penalty function described in (5) is used, then $\text{thresh}(x, \lambda, \alpha, \beta)$ is defined in (11). For other thresholding rules, such as soft-thresholding, hard-thresholding or NNG, the parameter α, β are irrelevant.

[7] These results are not included in Section 4 due to space limitations.

[8] A function $\Theta : \Re^N \longrightarrow \Re$ is said to be ρ-weakly convex if $\Theta(\mathbf{x}) = \frac{\tau}{2}\|\mathbf{x}\|_2^2 + p(\mathbf{x})$ is convex when $\tau \geqslant \rho \geqslant 0$. See [12] for details.

[9] We provide additional results for $R = \{18,\ 24,\ 30,\ 36\}$ in [25].

rule in Algorithm 4 are linearly varied for the first M iterations via $\alpha = \frac{k\cdot(\alpha_0-\alpha_M)}{M}+\alpha_0$ and $\beta = \frac{k\cdot(\beta_0-\beta_M)}{M}+\beta_0$ and then remain constant. We found experimentally that taking $M = 100$, $\alpha_0 = 0.95$, $\beta_0 = 0.05$, $\alpha_M \in [0.2, 0.35]$ and $\beta_M \in [0.65, 0.80]$ greatly improves our computational results in terms of better sparsity and MSE when compared to a fixed choice of α, β.



(a) PSNR vs time. $\lambda = 0.01$. (b) MSE vs. sparsity. $\lambda = 0.01$. (c) MSE vs. sparsity. $\lambda = 0.02$.

Fig. 1. Computational results for CSC applied to the Lena test image. The red and yellow curves are respectively for SPORCO [23] solving problem (1) with a dictionary of 144 non-separable ("ns") filters, and solving problem (6) with the corresponding separable ("s") approximation with 30 filters. The other curves are for Algorithm 4 applied to problems (4) and (6) with separable approximations of 24 and 30 filters, as indicated. When $\lambda = 0.01$ we provide (a) the evolution of the PSNR versus time as well as (b) the relationship between MSE and sparsity, while for $\lambda = 0.02$ we only show (c) the relationship between MSE and sparsity. See also Table 1. These results (PSNR evolution and MSE and sparsity relationship) are representative for the majority of images in the USC-SIPI database [24].

A sparsity measure is defined as $100 \cdot \frac{\|\mathbf{x}\|_0}{N}$, where $\mathbf{x}$ represents the coefficient maps and N is the number of pixels in the input image. We have observed experimentally that a sparsity measure of 65% gives a good compromise that balances sparsity, reconstruction error (MSE or SNR) and number of total iterations needed to reach the sparsity target for the CSC application. All implementations are run until the sparsity target is reached or until there is no significant change in the sparsity measure, which along with other statistics (PSNR, MSE, time, etc.) is collected every 10 iterations.

A comparison of the CSC performance of SPORCO solving problem (1) with a dictionary of 144 non-separable filters and Algorithm 4 solving problems (4) and (6) with dictionaries of 24 and 30 separable filters is presented in Fig. 1 and Table 1. Results for problem (6) solved with SPORCO with the dictionary of 30 separable filters are included to provide a performance baseline. With respect to Algorithm 4, solving problem (6) is both faster than solving problem (4) and gives solutions that are sparser, albeit with a higher reconstruction error. When compared to the state-of-the-art method, Algorithm 4 applied to problem (6) attains a competitive sparsity level while being 3∼5 times faster. Furthermore, when at least 30 separable filters are used to solve (6), the reconstruction error (SNR or MSE) is superior to that of the state-of-the-art.

Method	λ	Iter.	Time (sec.)	SNR (dB)	MSE $\times 10^{-5}$	Sparsity (%)
SPORCO [23]	0.005	600	40.91	37.51	0.6	80.28
	0.01	130	21.46	32.70	1.8	64.57
solves (1)	0.02	30	5.31	28.13	5.4	64.59
SPORCO [23]	0.005	100	3.21	33.02	1.76	68.18
	0.01	30	1.13	29.79	3.7	60.51
R=30, solves (6)	0.02	20	0.77	26.64	10.0	47.76
Algorithm 4	0.005	800	41.61	36.88	0.7	105.58
	0.01	400	20.76	33.98	1.4	101.72
R=24, solves (4)	0.02	300	15.63	30.76	2.9	67.43
Algorithm 4	0.005	800	15.93	34.13	1.3	63.10
	0.01	220	4.34	31.6	2.4	64.14
R=24, solves (6)	0.02	90	1.81	28.49	4.9	62.49
Algorithm 4	0.005	800	44.21	39.20	0.4	104.48
	0.01	400	22.09	35.87	0.9	102.99
R=30, solves (4)	0.02	300	16.62	31.88	2.2	68.97
Algorithm 4	0.005	800	19.79	36.51	0.7	67.85
	0.01	250	6.07	33.3	1.6	64.77
R=30, solves (6)	0.02	90	2.22	29.31	4.1	64.43

Table 1. Computational results when solving the case described in Fig. 1. for different values of λ.

5. CONCLUSIONS

We have proposed a computationally efficient algorithm to solve the CSC problem when the dictionary filters are separable. We consider two different formulations of this problem; one that explicitly computes the coefficients of the separable approximation of the original non-separable filter bank, and another that computes a reduced representation corresponding to the separable approximation with fewer filters. This algorithm, which applies the convolution in the spatial domain, is based on the FISTA/NIHT algorithm, and uses a non-convex penalty function, with a shrinkage rule that be interpreted as a synthesis between soft-thresholding and the Non-Negative Garrote. Our experiments indicate that this particular penalty function is key to the effectiveness of our proposed method. Overall our FISTA/NIHT-based method attains competitive performance when compared to the state-of-the-art.

6. REFERENCES

[1] J. Yang, K. Yu, and T. Huang, "Supervised translation-invariant sparse coding," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010, pp. 3517–3524.

[2] M. Zeiler, D. Krishnan, G. Taylor, and R. Fergus, "Deconvolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010, pp. 2528–2535.

[3] B. Wohlberg, "Efficient algorithms for convolutional sparse representations," *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 301–315, Jan. 2016.

[4] R. Rigamonti, A. Sironi, V. Lepetit, and P. Fua, "Learning separable filters," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013, pp. 2754–2761.

[5] A. Sironi, B. Tekin, R. Rigamonti, V. Lepetit, and P. Fua, "Learning separable filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 1, pp. 94–106, Jan 2015.

[6] Y. Nakatsukasa, T. Soma, and A. Uschmajew, "Finding a low-rank basis in a matrix subspace," *CoRR*, vol. abs/1503.08601, 2015.

[7] P. Rodriguez, "A two-term penalty function for inverse problems with sparsity constrains," Submitted to *European Signal Processing Conference*, Sept. 2017.

[8] L. Breiman, "Better subset regression using the nonnegative garrote," *Technometrics*, vol. 37, no. 4, pp. 373–384, Nov. 1995.

[9] S. Voronin and H. Woerdeman, "A new iterative firm-thresholding algorithm for inverse problems with sparsity constraints," *Applied and Computational Harmonic Analysis*, vol. 35, no. 1, pp. 151 – 164, 2013.

[10] M. Kowalski, "Thresholding rules and iterative shrinkage/thresholding algorithm: A convergence study," in *IEEE International Conference on Image Processing (ICIP)*, Oct. 2014, pp. 4151–4155.

[11] I. Selesnick and I. Bayram, "Sparse signal estimation by maximally sparse convex optimization," *IEEE Transactions on Signal Processing*, vol. 62, no. 5, pp. 1078–1092, March 2014.

[12] I. Bayram, "On the convergence of the iterative shrinkage/thresholding algorithm with a weakly convex penalty," *IEEE Transactions on Signal Processing*, vol. 64, no. 6, pp. 1597–1608, March 2016.

[13] B. Wohlberg, "Efficient convolutional sparse coding," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 7173–7177.

[14] A. Antoniadis, "Wavelet methods in statistics: some recent developments and their applications," *Statist. Surv.*, vol. 1, pp. 16–55, 2007.

[15] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[16] T. Blumensath and M. Davies, "Normalized iterative hard thresholding: Guaranteed stability and performance," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 298–309, April 2010.

[17] M. Zeiler, G. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *International Conference on Computer Vision (ICCV)*, 2011, pp. 2018–2025.

[18] R. Chalasani, J. C. Principe, and N. Ramakrishnan, "A fast proximal method for convolutional sparse coding," in *International Joint Conference onNeural Networks (IJCNN)*, Aug. 2013, pp. 1–5.

[19] H. Bristow, A. Eriksson, and S. Lucey, "Fast convolutional sparse coding," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013, pp. 391–398.

[20] F. Heide, W. Heidrich, and G. Wetzstein, "Fast and flexible convolutional sparse coding," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 5135–5143.

[21] B. Wohlberg, "Boundary handling for convolutional sparse representations," in *IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, Sept. 2016, pp. 1833–1837.

[22] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan. 2011.

[23] Brendt Wohlberg, "SParse Optimization Research COde (SPORCO)," Software library available from `http://purl.org/brendt/software/sporco`, 2016.

[24] "USC-SIPI image database," Available from http://sipi.usc.edu/database/.

[25] P. Rodriguez, "Convolutional sparse coding with separable filters simulations," Matlab library available from `http://sites.google.com/a/istec.net/prodrig/Home/en/pubs/csrSep`, 2016.

[26] P. Rodriguez, "Alternating optimization low-rank expansion algorithm to estimate a linear combination of separable filters to approximate 2d filter banks," *IEEE Asilomar Conference on Signals, Systems and Computers*, Nov. 2016.

[27] T. Kolda and B. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.