UNIVERSITY OF BRISTOL

Mayo, P., Karakuş, O., Holmes, R., & Achim, A. (2020). Representation Learning via Cauchy Convolutional Sparse Coding. Unpublished. https://arxiv.org/abs/2008.03473

Early version, also known as pre-print

Link to publication record in Explore Bristol Research
PDF-document

University of Bristol - Explore Bristol Research
General rights

Representation Learning via Cauchy Convolutional Sparse Coding

Perla Mayo, Oktay Karaku, Robin Holmes, and Alin Achim,

Abstract

In representation learning, Convolutional Sparse Coding (CSC) enables unsupervised learning of features by jointly optimising both an ℓ_2-norm fidelity term and a sparsity enforcing penalty. This work investigates using a regularisation term derived from an assumed Cauchy prior for the coefficients of the feature maps of a CSC generative model. The sparsity penalty term resulting from this prior is solved via its proximal operator, which is then applied iteratively, element-wise, on the coefficients of the feature maps to optimise the CSC cost function. The performance of the proposed Iterative Cauchy Thresholding (ICT) algorithm in reconstructing natural images is compared against the common choice of ℓ_1-norm optimised via soft and hard thresholding. ICT outperforms IHT and IST in most of these reconstruction experiments across various datasets, with an average PSNR of up to 11.30 and 7.04 above ISTA and IHT respectively.

Index Terms

Cauchy-based penalty function, convolutional sparse coding, proximal splitting

I. Introduction

Representation learning seeks to understand the underlying patterns and structures that give raise to the data of interest. This often involves using generative models to describe the processes involved in the formation of this data, using known or assumed priors [1]. In some of these models it is assumed the data arises from of a linear operation among the elements of a set of basic or canonical features. Thus, in addition to selecting this set of features, it is also necessary to obtain their respective coefficients for generating any given sample. Computing the coefficients for such a sample effectively transforms it into the chosen feature domain. This transformation process is referred as encoding, whilst decoding corresponds to the reverse action of transforming back into the original domain [1], [2].

Establishing an effective choice of features for the generative model can aid in understanding the nature of the data. Further, the resulting coefficients after encoding can be used in-place of the raw data for many tasks such as image compression [3]–[5], image denoising [6], [7], image super-resolution [8], [9]; or even image classification [10], [11] or anomaly detection [12] for those requiring some discriminative power in them.

Thus, determining a suitable feature set for the data in question is a crucial task, but how should this be accomplished? Early on, it was common to employ a set of predefined or fixed basis features. Sets such as Wavelets and ones obtained from the Discrete Cosine Transform have been used with success for image denoising and compression respectively. For instance, to denoise images, a common practice is to transform them to the wavelet domain, in which a threshold can be applied to the wavelet coefficients [7]. Reversing the transform to the original domain after this thresholding then results in a cleaner image. However, feature sets such as these are in some sense universal. As such they are not always effective in capturing specific traits of a particular dataset. This can sometimes result in vital particularities of the data being lost. For this reason different approaches enabling the unveiling of new meaningful data specific information have become more prevalent.

As previously mentioned, the assumptions made will guide the design of the model to utilise for these purposes. Principal component analysis (PCA) [13] and independent component analysis (ICA) [14] provide the means to determine the underlying components comprising the data of interest. The difference between these two being that ICA makes a further assumption regarding the independence among these components. In either case, the generative model corresponds to a dot product that meets some orthogonality conditions on the (squared) matrix of features. There is no strict requirement however, for the feature matrix to be square. For instance, in Autoencoders (AE) [15] the goal is to train a network in an unsupervised fashion such that its weights both encode and decode the data in a lower dimensional space. Alternatively, there is dictionary learning and sparse coding [16], in which it has been suggested that overcomplete sets are capable of describing as well as (if not better than) complete ones as they are able to unveil a bigger number of underlying features [16]. Since the features belong to an overcomplete matrix, the model to solve is underdetermined and an infinity of solutions becomes available. This can be remedied by assuming the data representation is sparse, meaning that only a few elements of the feature matrix take part in the formation of the data and hence most of the elements in the vector of coefficients are set to zero. This is modelled by the addition of a penalty term known to enforce this behaviour. The assumption of sparsity has been motivated by the way in which the V1 cells from the visual cortex work [17].

With this addition the model not only learns the features to represent the data but also the coefficients describing their contribution. This is commonly achieved by splitting the learning into two tasks, i.e., a step is devoted to learn the elements of the dictionary and the other to learn the elements in the vector of coefficients. There is evidence showing that the latter step can be the most critical for the model to succeed in the representation task [2]. In [2] the power of encoding was demonstrated regardless the choice of learning (or lack of it) for the features. This motivates efforts on the design of novel approaches aiming to learn the coefficients in the encoding stage of the algorithm.

The core contribution of this work is the derivation and effective demonstration of a new regularisation term used during the encoding step of Convolutional Sparse Coding (CSC). This term arises from the assumption that feature map coefficients follow a Cauchy distribution. To make use of this new regularisation we propose the Cauchy proximal operator, which when implemented iteratively follows in the vein of shrinkage algorithms [18]–[20] and gives raise to an algorithm, which we refer to as iterative Cauchy thresholding (ICT). Unlike existing previous approaches this algorithm does not perform explicit

thresholding, resulting in values approaching 0 but not necessarily locking to it. The power of this new regularisation is shown on a reconstruction task for 2D images, and evaluated against the common choices of soft and hard thresholding algorithms.

The remaining of this manuscript is organised as follows. The backbone and derivation of the proposed algorithm is reviewed in detail in section III along with related work that inspired our approach. In section IV the algorithm used for the reconstruction task is shown. The experiments conducted are found in section V along with their results. Lastly, section VI offers a discussion, conclusion, and future lines of work.

II. THEORETICAL PRELIMINARIES

In a basic generative model, it is assumed the observations $\mathbf{y} \in \mathbb{R}^M$ can be estimated from a linear combination of the column vectors (also referred as atoms, codes or features) of the dictionary matrix $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_N] \in \mathbb{R}^{M \times N}$. The contribution of each one of these elements is given by the coefficients in $\mathbf{x} \in \mathbb{R}^N$ such that there is one coefficient per column in $\mathbf{A}$:

$$\hat{\mathbf{y}} = \mathbf{A}\mathbf{x} \tag{1}$$

Since $\mathbf{y} \approx \hat{\mathbf{y}}$, there exists a vector ϵ such that $\epsilon = \mathbf{y} - \hat{\mathbf{y}}$ or, equivalently

$$\mathbf{y} = \hat{\mathbf{y}} + \epsilon \tag{2}$$

with $\hat{\mathbf{y}}, \epsilon \in \mathbb{R}^M$. For dictionary learning and sparse coding, the dictionary matrix $\mathbf{A}$ is overcomplete $N > M$. In addition, there is a one-to-one spatial correspondence between the features and the data to reconstruct, i.e. the dimension of the features has to be the same as that of the data, which can be impractical for high dimensional signals. This can be alleviated by using patches extracted from the original signal instead, reducing the dimension of the dictionary atoms to the one of these patches. Thus, the observations $\mathbf{y}_i$ corresponds to patches extracted from the original signal of original dimension P and only L atoms participate in the generation of the data ($L \ll N$). By using patches it becomes necessary to perform pre- and post-processing of the data to extract the patches and then bring them together to reconstruct each sample. For this to work, it is assumed these patches are independent, even if they come from the same sample, which is not necessarily true. There are two main ways to extract patches from the data, one of them is restricting them to not overlap. This, in addition to the independence assumption, is later on reflected in blocking artifacts when the samples are reconstructed. On the other hand, when the patches are overlapped, an average is performed, which also results in a degraded version of the original sample as there is now a smoothing effect present in them. Furthermore, the learned features are often translated versions of other atoms within the set (they are not shift invariant).

The use of the convolution operator in the generative model helps to address the aforementioned limitations of dictionary learning [21], [22]. Thus, it evolves to Convolutional Sparse Coding (CC). Eq. 3 describes this model.

$$\hat{\mathbf{y}} = \sum_{k=1}^{K} \mathbf{f}_k * \mathbf{z}_k \tag{3}$$

TABLE I
PENALTY TERMS THAT PROMOTE SPARSITY AND THEIR PROXIMAL OPERATORS

Algorithm	Penalty term	Optimising function
IHT	$|x_i|^0$	$x_i = \begin{cases} x_i, & |x_i| > \lambda \\ 0, & |x_i| \leq \lambda \end{cases}$
IST	$|x_i|$	$x_i = \begin{cases} x_i - \lambda/2, & x_i > \lambda/2 \\ x_i + \lambda/2, & x_i < -\lambda/2 \\ 0, & otherwise \end{cases}$
ILT	$\lambda \log(\delta + x_i)$	$x_i = \begin{cases} \frac{1}{2}\left((x - \delta) \pm \sqrt{(x+\delta)^2 - 2\lambda} \right), & x \geq \sqrt{2\lambda} - \delta \\ \frac{1}{2}\left((x + \delta) \pm \sqrt{(x-\delta)^2 - 2\lambda} \right), & x \geq \sqrt{2\lambda} + \delta \end{cases}$

where the signal of interest $\mathbf{y} \in \mathbb{R}^P$ is now modelled as a sum of K filters $\mathbf{f}_k \in \mathbb{R}^M$ convolved with their respective feature map $\mathbf{z}_k \in \mathbb{R}^Q$, with $P = M + Q - 1$ for $k = 1, 2, ..., K$. Note that $\mathbf{y}$ is the complete original signal. The extension to higher dimensional data is straightforward. Nevertheless, for ease of reading the equations are expressed purely using one dimensional data.

In either of the two mentioned generative models, the learning of the features can be done by minimising the error between the estimated and the observed data. For instance, for CSC:

$$
\begin{aligned}
\mathbf{f}^* &= \arg \min_{\mathbf{f}} \mathcal{L}(\mathbf{f}, \mathbf{z}) \\
&= \arg \min_{\mathbf{f}} ||\mathbf{y} - \hat{\mathbf{y}}||_2^2 \\
&= \arg \min_{\mathbf{f}} ||\mathbf{y} - \sum_{k=1}^{K} \mathbf{f}_k * \mathbf{z}_k||_2^2
\end{aligned}
\tag{4}
$$

where $\mathbf{f} = [\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_K]$ and $\mathbf{z} = [\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_K]$. In dictionary learning the optimisation would be carried over the matrix $\mathbf{A}$. From now on, the generative model considered in the paper is the CSC.

To seek for sparsity, it suffices to add a regularisation term to the optimisation function as

$$
\begin{aligned}
\mathbf{f}^*, \mathbf{z}^* &= \arg \min_{\mathbf{f}, \mathbf{z}} \mathcal{G}(\mathbf{f}, \mathbf{z}) \\
&= \arg \min_{\mathbf{f}, \mathbf{z}} \mathcal{L}(\mathbf{f}, \mathbf{z}) + \lambda \varphi(\mathbf{z}) \\
&\text{s.t. } ||\mathbf{f}_k||_2 = 1, k = 1, 2, ..., K
\end{aligned}
\tag{5}
$$

in which it is now required to learn, in addition to the set of features in $\mathbf{f}$, its coefficients in the feature maps $\mathbf{z}_k$. The constraint on the filters prevent it to absorb most of the energy during the learning.

The learning is carried on by iteratively alternating the optimisation of cost function over $\mathbf{f}$ and $\mathbf{z}$. This means that in a first step (z-step), the cost function will be minimised by assuming $\mathbf{f}$ is fixed. The opposite happens during the f-step.

Several approaches have emerged aiming to solve Eq. 5 w.r.t. $\mathbf{f}$ to learn features from the data in addition to Gradient Descent, such as K-SVD [23] and the more image statistic-adapted SparseDT [24]. Similarly, optimising the cost function w.r.t. $\mathbf{z}$ will result in the learning of the sparse coefficients. Such optimisation

depends on the choice of penalty function. If one is to seek for the sparsest solution, then the penalty term chosen is the ℓ_0-norm. Hence, finding the set of coefficients that optimise Eq. 5 is a combinatorial (NP-hard) problem. Broadly speaking, there are two main approaches to solve said regularisation term: greedy and relaxed algorithms. The first category focuses on solving the ℓ_0-norm whilst the second one considers its relaxed version (the ℓ_1-norm). For the former, Matching Pursuit is one of the most common solvers in which coefficients are chosen one by one in a greedy fashion until a stopping criteria is met. If, on the other hand, one chooses the ℓ_1-norm (LASSO), the function to optimise is now non-smooth convex.

In these circumstances, the choice of penalty term is based on the known behaviour (shape) of the function. Thus, as long as one knows the function has a shape that can enforce sparsity, such function can be used for $\varphi(\cdot)$. Some alternatives are the non-convex ℓ_p-norm (with $p \leq 1$) or the (also non-convex) log regulariser among others. A comprehensive review of these sparsity-enforcing functions can be found in [25] and references therein.

Iterative algorithms have come along with an associated sparsity enforcing penalty term. IST is often involved when the function to optimise makes use of the ℓ_1-norm, IHT for the ℓ_0-norm and recently the use of the iterative log thresholding (ILT) [20] has been proposed to optimise the log regulariser. Table I summarises the equations involved in these algorithms.

IST and IHT can also be derived via surrogate functions in which one seeks to separate the terms involved in the cost function. Regardless of the chosen algorithm to use, these thresholding operators are applied in an element-wise fashion. These three approaches suppress any value below some threshold but it is only IST and ILT that update values higher than said threshold. In the case of IST this has a direct impact on the results as they often exhibit blurring.

III. ITERATIVE CAUCHY THRESHOLDING

The Cauchy assumption in the field of image processing is not new as it has previously been used to model the noise corrupting the images of interest [26], [27]. Nonetheless, in this work it is not the noise but the coefficient values involved in the generative models the ones that are assumed to follow this distribution. In fact, the assumption of a Cauchy prior for the model has been done with success in the past [28]–[32].

The encoding step depends on the regularisation term in the optimisation model. This term could fall into the non-smooth convex functions, such as the ℓ_1-norm; non-smooth non-convex, such as the log regulariser or the ℓ_0-pseudo norm; or smooth non-convex penalty terms, such as the one explored in this paper. This function is derived from a statistical assumption on the coefficients, serving as prior in a maximum a posteriori (MAP) approach. The resultant learning algorithm corresponds to a function which, despite the non-convexity of its regularisation term, is guaranteed to convergence under a certain condition. Specifically, it is the Cauchy distribution the one assumed to drive the learning framework. The use of this prior enables the learning of the coefficients by iteratively applying its proximal operator on the coefficients, achieving shrinkage around a non-explicit threshold that emerges naturally from the equations involved in this process. In addition, the parameters shaping the distribution of the coefficients can be estimated from the observations, facilitating the use of this method.
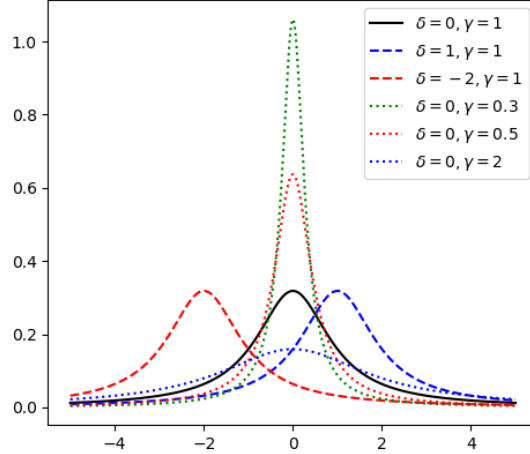
Fig. 1. Cauchy PDF with different values for the parameters δ and γ

A. *The Cauchy distribution*

The Cauchy distribution belongs to the family of the Symmetric α-Stable (SαS) distribution. Its location and dispersion are described by the parameters δ and γ respectively, and its p.d.f. is defined by

$$p(x) = \frac{\gamma}{\pi(\gamma^2 + (x - \delta)^2)} \tag{6}$$

whilst Figure 1 illustrates their role on the distribution.

For the aim of the proposed work (enforcing sparsity), it is required that $\delta = 0$, which in turn simplifies the expression to work with. On the other hand, the parameters of the distribution can be estimated from the data itself using maximum likelihood estimation as

$$\gamma = \arg\min_{\gamma} - \sum_{t=1}^{T} \log(p(x) + \epsilon) \tag{7}$$

where ϵ is a very small value.

The overall cost function to be optimised is composed of two functions, as illustrated in Eq. 5. The data fidelity term $\mathcal{L}(\cdot, \cdot)$ being convex with $\varphi(\cdot)$ possibly non-smooth and/or non-convex. In proximal splitting, the functions that present challenges during conventional optimisation techniques are projected into a convex set via their proximal operators. Note that the number of functions involved in the optimisation can be ≥ 2. The proximal operator of a given function $\varphi(\cdot)$ can be obtained by solving:

$$\text{prox}_{\lambda\varphi}(x) = \arg\min_{\mathbf{z}} \ (z - x)^2 + \lambda\varphi(z) \tag{8}$$

B. *The Cauchy Proximal Operator*

Following a MAP approach, the penalty term on the coefficients in $\mathbf{z}$ is then defined as $\varphi(\cdot) = -\log(p(\cdot))$, with $p(\cdot)$ as defined in Eq. 6 and setting $\delta = 0$. This penalty term is applied individually on

every element of **z**. Thus, plugging in this into Eq. 8 it is possible to derive the Cauchy proximal operator as

$$\text{prox}_{\lambda\varphi}(x) = \arg\min_{\mathbf{z}} \quad (z-x)^2 + \lambda\varphi(z)$$

$$= \arg\min_{\mathbf{z}} \quad (z-x)^2 - \lambda\log\left(\frac{\gamma}{\pi\left(\gamma^2+z^2\right)}\right) \tag{9}$$

Taking the derivative to find the stationary points:

$$\frac{d}{dz}\left((z-x)^2 - \lambda\log\left(\frac{\gamma}{\pi\left(\gamma^2+z^2\right)}\right)\right) = 0$$

$$\frac{d}{dz}\left((z-x)^2\right) - \lambda\frac{d}{dz}\left(\log\left(\frac{\gamma}{\pi\left(\gamma^2+z^2\right)}\right)\right) = 0$$

$$2(z-x) - \lambda\frac{d}{dz}\left(-\log\left(\gamma^2+z^2\right)\right) = 0$$

$$2(z-x) + \lambda\frac{d}{dz}\left(\log\left(\gamma^2+z^2\right)\right) = 0$$

$$2(z-x) + \lambda\frac{1}{\left(\gamma^2+z^2\right)}\frac{d}{dz}\left(\gamma^2+z^2\right) = 0$$

$$2(z-x) + \lambda\frac{2z}{\left(\gamma^2+z^2\right)} = 0$$

$$(\gamma^2+z^2)(z-x) + \lambda z = 0$$

Lastly, rearranging the terms, we get

$$z^3 - xz^2 + (\gamma^2+\lambda)z - \gamma^2 x = 0 \tag{10}$$

Using the Cardano's method to find the roots of the previous cubic equation with $a = 1$, $b = -x$, $c = \gamma^2 + \lambda$ and $d = -\gamma^2 x$, one finally gets to:
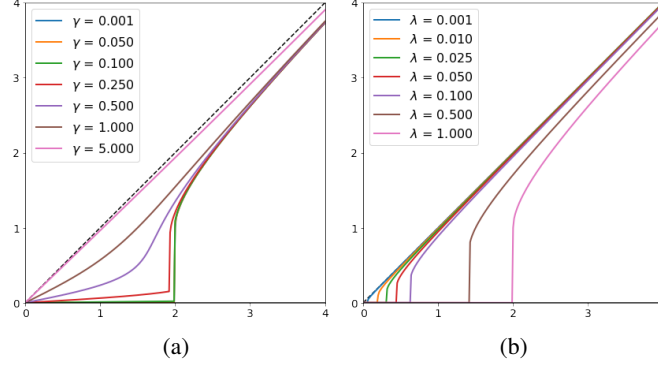
$$z = \frac{x}{3} + t \tag{11}$$

where

Fig. 2. Behaviour of ICT for varying (a) γ's and (b) λ's.

$$t = \sqrt[3]{-\frac{q}{2} + \sqrt[2]{\Delta}} + \sqrt[3]{-\frac{q}{2} - \sqrt[2]{\Delta}}$$

$$\Delta = \frac{q^2}{4} + \frac{p^3}{27}$$

$$p = \lambda + \gamma^2 - \frac{x^2}{3}$$

$$q = -\frac{2}{27}x^3 + \frac{1}{3}\left(\lambda - 2\gamma^2\right)x$$

The Cauchy proximal operator, thus, requires to choose values for the parameters γ and λ, for which it becomes ideal to understand their function in the operator. By fixing γ to a specific value and then vary λ and vice-versa it is possible to gain an intuition of their roles. In fact, it is found that the value of γ shapes the thresholding function and smaller values contribute to a more aggressive shrinkage near the threshold, whilst λ shifts the threshold location. Fig. 2a and 2b illustrate this behaviour.

In fact, when $\gamma \to 0$ the threshold $\to 2\lambda$ and the shape of the function approximates the ILT. On the other hand, when $\lambda \to 0$, the roots of Eq. 10 $\to \gamma i, -\gamma i$, and x, which would keep the values unchanged, i.e., not shrinkage would be performed. Nonetheless, in this work we do not treat γ as a tunable parameter. Instead, this value is estimated from the data following the approach mentioned in Section III-A.

One could compare the Cauchy and log penalty terms (third row in Table I) since both are shaped by the logarithm function and some parameter, δ for ILT and γ for ICT. The corresponding proximal operators are considerably different. A major difference between ICT and the rest of the algorithms presented in this manuscript so far is the lack of an explicit threshold. The coefficents are still shrunk according to the proximal operator in an iterative manner, reaching values closer to zero but not necessarily locking on it.

The penalty term derived from the Cauchy distribution is a smooth non-convex function, which makes the optimisation of $\mathcal{G}(\cdot, \cdot)$ challenging . However, the cost function defined in Eq. 5, as a whole, can be guaranteed to converge to a global minimum, if the following condition is met [33]:
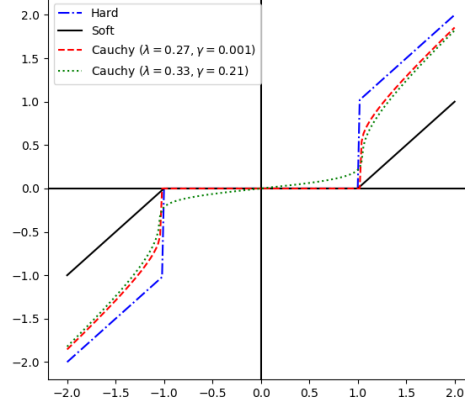
$$\lambda \leq 8\gamma^2 \tag{12}$$

Fig. 3. Behaviour of the different thresholding algorithms and, in the case of Cauchy, using different parameters.

Specifically, the condition given in Eq. 12 ensures that the cost function in the Cauchy proximal operator (9) converges. This condition guarantees convergence in the scenario in which the proximal operator needs to be applied in an iterative manner for inverse problems [33] It is this iterative process the one that gives rise to our proposed ICT algorithm, whose pseudocode is presented in Algorithm 1. Note that an additional parameter η is present as it accounts for the learning rate, thus, the original equation contains $\eta\lambda$, and since $\lambda = 1$ the algorithm has η only, which also affects the convergence condition to $\eta \leq 8\gamma^2$ instead.

Algorithm 1 Iterative Cauchy Thresholding

Initialise $\mathbf{x}$ with 0's
Set η, γ and λ
Choose stopping criterion. In this work this corresponds to a max number of iterations
while Stopping criteria has not been met **do**
    Compute $\mathbf{z} \leftarrow \mathbf{z} - \eta\nabla_{\mathbf{z}}\mathcal{L}(\mathbf{f}, \mathbf{z})$
    Shrink every element in $\mathbf{z}$ using Eq. (11).
end while

This condition is easily applied when the generative model corresponds to CSC. As shown by [33], this arises from the condition being derived by taking the second derivative where the generative model is no longer involved and the resultant expression is dependant only on the hyper-parameters.
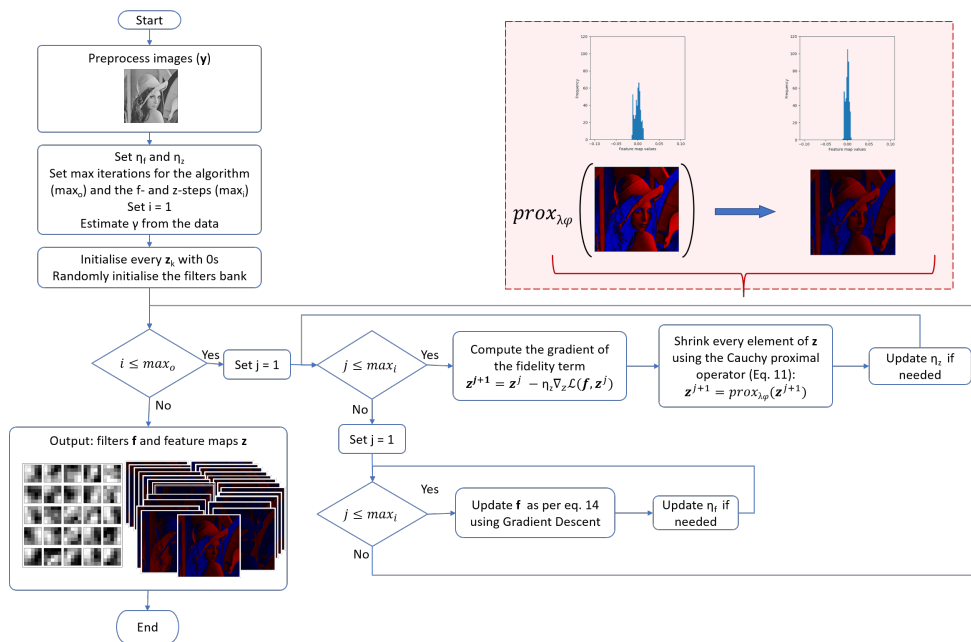
Fig. 4. Block diagram of the Cauchy Convolutional Sparse Coding algorithm. After a few iterations the coefficients within the feature maps get closer to zero.

IV. CAUCHY CONVOLUTIONAL SPARSE CODING

In this section, our new CSC algorithm for representation learning is introduced. It is based on the use of the Cauchy proximal operator through an iterative process in order to encode the data for the z-step. The cost function is derived via MAP. The prior knowledge employed and which then translates into the penalty function corresponds to the assumed statistical distributions of the coefficients [22].

By using the Cauchy distribution in the generative model it is now required to perform the z-step via the Cauchy proximal operator. Our goal is thus to solve Eq. (5) for the feature maps using the Cauchy penalty function. Specifically, the cost function is now:

$$\mathcal{G}(\mathbf{f}, \mathbf{z}) = ||\mathbf{y} - \hat{\mathbf{y}}||_2^2 - \lambda \sum_{k=1}^{K} \sum_{q=1}^{Q} \log \left(\frac{\gamma}{\pi(\gamma^2 + z_{k,q}^2)} \right) \tag{13}$$

with $\hat{\mathbf{y}}$ as defined in Eq. 3. Thus, the full algorithm aims to learn the set of filters $\mathbf{f}$ and the feature maps $\mathbf{z}$ associated to the data from a dataset of size T. Note that extending the cost function defined in 5 to learn from more than one sample (i.e. dataset size > 1) is straightforward and hence not detailed in here.

As it is common in similar algorithms, the proposed approach works by alternating between the learning of the features and the learning of the coefficients, until a convergence criterion is met. This can consist in reducing the reconstruction error below some predefined value or in a maximum number of iterations to be reached. Gradient descent is used as learning approach for the features (f-step) in conjunction with a chosen thresholding algorithm for the coefficients (z-step). In both learning steps the learning rate adapts so that overshooting over the local minima is prevented. This is achieved by halving the learning rate for the current step following an observed increase in value subsequent to an update.

The f-step is solved by minimising Eq. 13 over $\mathbf{f}$, which can be written compactly as

---

Algorithm 2 Cauchy Convolutional Sparse Coding

Initialise $\mathbf{z}_k$ with 0's, $k = 1, 2, .., K$
Initialise randomly $\mathbf{f}_k$, $k = 1, 2, .., K$
Estimate γ from the data using Eq. 7
Choose stopping criteria
while Overall stopping criteria has not been met **do**
   Set $\mathbf{z}^{old} \leftarrow \mathbf{z}$
   while Stopping criteria for z-step has not been met **do**
      Set $C_O \leftarrow \mathcal{G}(\mathbf{f}, \mathbf{z})$
      For every $\mathbf{z}_k$ compute:
      $\mathbf{z}_k \leftarrow \mathbf{z}_k - \eta_z \nabla_{\mathbf{z}_k} \mathcal{L}(\mathbf{f}, \mathbf{z})$
      Shrink $\mathbf{z}_k$ using Eq. (11).
      Set $C_N \leftarrow \mathcal{G}(\mathbf{f}, \mathbf{z})$
      if $C_N > C_O$ **then**
         Set $\mathbf{z} \leftarrow \mathbf{z}^{old}$
         Set $\eta_z = \eta_z/2$
      else
         Set $\mathbf{z}^{old} \leftarrow \mathbf{z}$
      end if
   end while
   while Stopping criteria for f-step has not been met **do**
      Set $C_O \leftarrow \mathcal{G}(\mathbf{f}, \mathbf{z})$
      For every $\mathbf{f}_k$ compute GD on the filers:
      $\mathbf{f}_k \leftarrow \mathbf{f}_k - \eta_f \nabla_{\mathbf{f}_k} \mathcal{L}(\mathbf{f}, \mathbf{z})$
      if $C_N > C_O$ **then**
         Set $\mathbf{f} \leftarrow \mathbf{f}^{old}$
         Set $\eta_f = \eta_f/2$
      else
         Set $\mathbf{f}^{old} \leftarrow \mathbf{f}$
      end if
   end while
end while

$$\mathbf{f}^* = \arg \min_{\mathbf{f}} \quad ||\mathbf{y} - \sum_{k=1}^{K} \mathbf{f}_k * \mathbf{z}_k||_2^2 \tag{14}$$

This requires taking the gradient over $\mathbf{f}$ and choosing a step size η_f for updating the features iteratively. This guarantees convergence since it involves the optimisation of the ℓ_2-norm, which is a smooth convex function.

Fig. 5. Reconstructions of image Lena (top row) using the algorithms (a) IHT, (b) IST and (c) ICT and the filters learned using (a) IHT, (b) IST and (c) ICT.
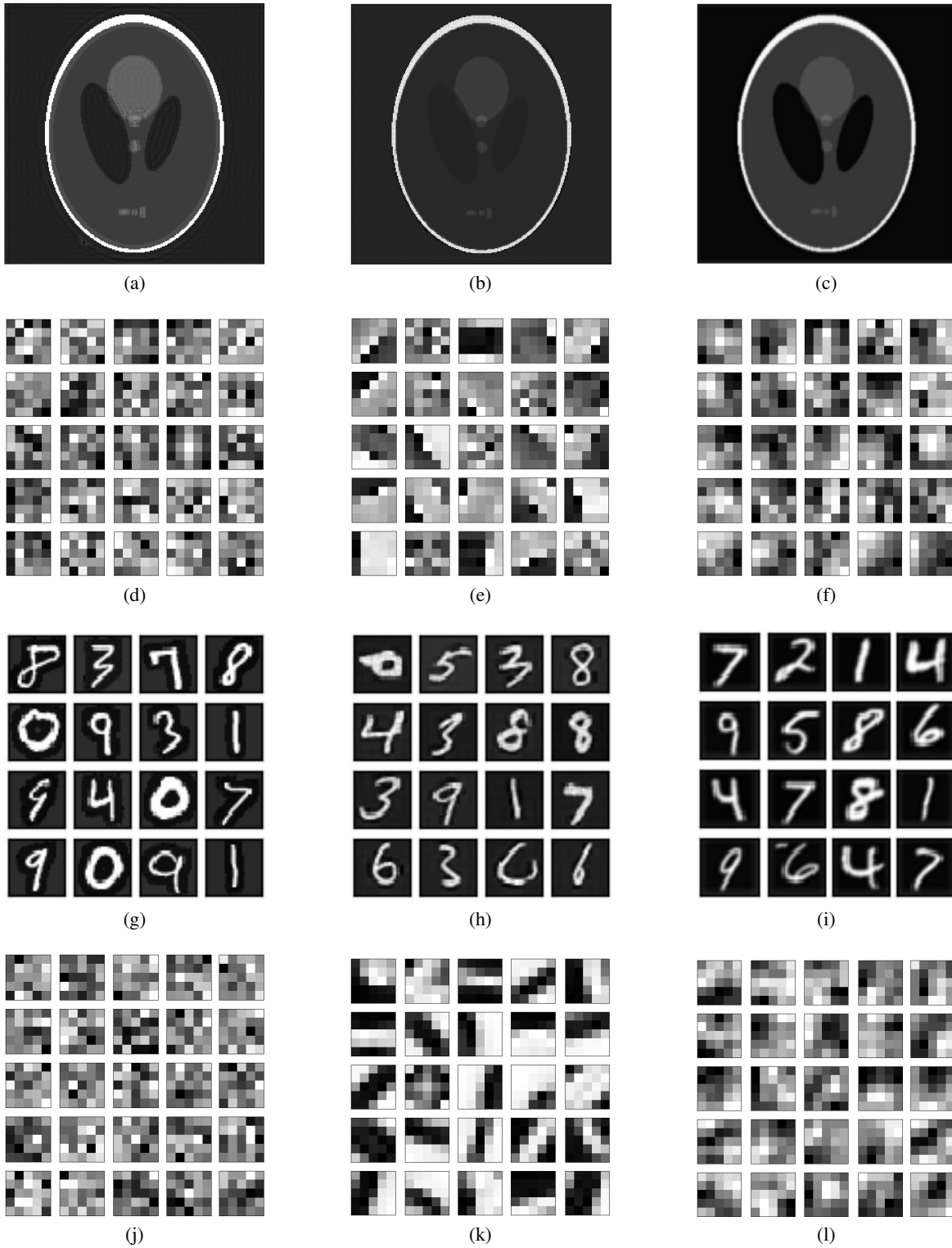
Fig. 6. Reconstructions of image Shepp-Logan Phantom (top row) using the algorithms (a) IHT, (b) IST and (c) ICT and the filters learned using (a) IHT, (b) IST and (c) ICT.

The key to implementing the Cauchy-CSC (CCSC) method consists in using ICT in the encoding phase of the algorithm. This is achieved by solving:

$$
\begin{aligned}
\mathbf{z}^* = \arg\min_{\mathbf{z}} \quad & ||\mathbf{y} - \sum_{k=1}^{K} \mathbf{f}_k * \mathbf{z}_k||_2^2 \\
& - \lambda \sum_{k=1}^{K} \sum_{q=1}^{Q} \log\left(\frac{\gamma}{\pi(\gamma^2 + z_{k,q}^2)}\right)
\end{aligned}
\tag{15}
$$

In addition to the regularisation parameter λ, one requires also to choose a learning rate η_z. Similarly to what has been done for the f-step, η_z is updated whenever the cost function increases as result of the previous coefficient update. The pseudocode of the whole approach is presented in Algorithm 2 and its diagram is depicted in figure 4.

V. SIMULATION RESULTS

In order to assess the performance of CSC when used in conjunction with ICT, IHT and IST we conducted a number of experiments. In particular, we focused on the reconstruction of 2D images in order to quantify the results of the said algorithms. The data employed were classical images such as Lena and the Shepp-Logan phantom, as well as the MNIST and AT&T faces[1] datasets. Before applying the representation learning algorithm, independently of the regulariser used, the data have been pre-processed to make them zero-mean. There is no pre-processing done to enforce the dataset to have unit variance since this could affect the estimation of the γ parameter required by the ICT algorithm. Since MNIST and the faces dataset are considerably large, a sample composed of $T = 500$ and $T = 30$ random images therein were used in the respective experiments.

The complete approach was performed 100 times for each dataset using different random initialisation for the filters. For the MNIST and AT&T datasets a random set of samples was also chosen at the beginning of each of their experiments. The maximum number of iterations was fixed to 100 per experiment.

Note that the hyperparameter λ incorporates the learning rate for IHT and IST, whereas for ICT it was set to 1 in order to leave it as close to the original cost function derived from MAP as possible. Hence, only the estimation of γ is required.

For ICT, the learning rate needs to meet the condition in Eq. 12. The additional tunable parameters employed were $K = 25$ and a filter size of 5×5 for all the experiments.

For an initial qualitative assessment, Figures 5 and 6 show the filters learned for the different datasets, along with the reconstructed images. We show samples from the experiments with the highest PSNR for each algorithm. By visually inspecting Figures 5 and 6, it can be seen that ICT can learn more meaningful filters since they seem to present less random patterns, in contrast to IST and IHT, in which some of these bases failed to be updated. In fact, we noticed that the initialisation of the filters plays an important role in their learning as sometimes there seem to be no learning at all for IHT as the filters have a noisy

---

[1]https://git-disl.github.io/GTDLBench/datasets/att_face_dataset/

TABLE II
PSNR RESULTS OF CSC USING DIFFERENT PENALTY TERMS

Dataset	ICT	IHT	IST
MNIST	20.36	**21.09**	18.57
AT&T Faces	**25.82**	25.31	20.37
Phantom	**24.20**	20.67	20.37
Lena	**32.39**	25.35	21.09

TABLE III
PROPORTION OF ZERO COEFFICIENTS LEARNED VIA CSC USING DIFFERENT PENALTY TERMS

Dataset	ICT	IHT	IST
MNIST	99.99	5.65	**1.16**
AT&T Faces	99.99	2.91	**1.65**
Phantom	100	2.60	**1.37**
Lena	99.99	**1.57**	1.60

appearance. This is in spite of their relatively good reconstruction performance with high PSNR values achieved and this confirms the dependence of reconstruction performance on the encoding step [2].

The performance of the three representation learning approaches is also assessed through quantitative analysis. The PSNR values for the reconstruction of each sample was computed and their average values are reported in Table II and Fig. 8 showing their respective boxplots. Table III reports the average proportion of non-zero elements in the learned feature maps. In both table II and table III, the best performance for each dataset is shown in bold.

Lastly, in Fig. 9, a plot of the learning performance as function of average PSNR as the iterations progress is provided.

From Fig. 9 we can see that ICT and IST reach the plateau corresponding to the highest PSNR early in the learning process, with IHT reaching its own maximum a few iterations later. It is ICT, however, the one that achieves the highest PSNR and requires the least iterations. Both IHT and IST requires tuning of a number of parameters for optimal performance, whereas for ICT the parameter γ is estimated directly from the data. In fact, the use of the iterative Cauchy algorithm requires the choice of only two values, the learning rate and the scale parameter. As noted in section III-B, γ can be estimated from the original data whilst η_z needs to obey the condition in Eq. 12.

From Table II we can see that ICT provides the best PSNR performances in three out of four cases, which is consistent with the visual evaluation. IST, on the other hand, is the one with the worst reconstruction performances, although it leads to the sparsest representations and the learning of seemingly sharper features in comparison to ICT. IST presents more consistency in regard of the PSNR results obtained as the inter-quartiles range is shorter than the other two algorithms, as seen in Fig. 8.

There is an increase of the average PSNR as the images increase in size for the three thresholding algorithms considered, with ICT exhibiting the highest of such jumps. Indeed, ICT performed better as the
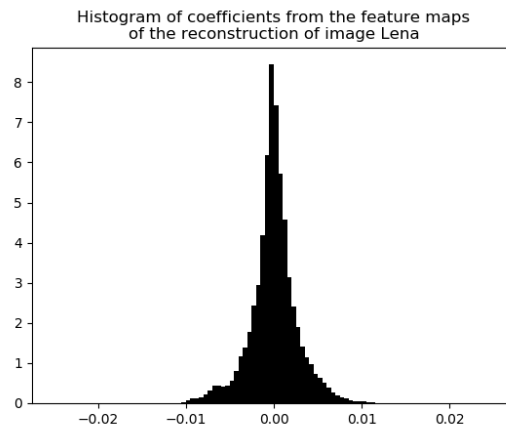
Fig. 7. Histogram of coefficient values from the 25 feature maps involved in the reconstruction of the image Lena using ICT. Y axis scale factor: 10^5.

size of the images increased, having very similar performance to IHT for the small size MNIST (28×28) dataset as opposed to the case of the the Shepp-Logan phantom (256×256) and Lena (512×512) images. Furthermore, despite the high PSNR values obtained with IHT, some degradation in the reconstructed images surrounding the edges and the lose of details is apparent. In the case of IST, the images exhibit a smoothing effect regardless of their dimension. Lastly, the reconstructed images produced by ICT also present some artifacts near the edges, which become more apparent in smaller image sizes.

With respect to Table III, it is evident that ICT is the approach that offers the least sparse solutions. Having a closer look to the histograms of coefficients (Fig. 7) it can be observed that most coefficients are in a very close vicinity of 0, which might explain the ability to learn most of the features most of the times whilst reducing their noisy appearances.

VI. CONCLUSIONS AND FUTURE WORK

In this work a new convolutional sparse coding framework based on a Cauchy model assumption is proposed. This approach enables the learning of filters and their respective feature maps by using said distribution as prior for the coefficients in the latter ones, which results in a new cost function. The Cauchy proximal operator was derived and used to optimise it and this requires a preliminary step before the learning process, which involves the estimation of the corresponding scale parameter. The performance was evaluated on four different datasets and compared against the reconstruction performance achieved using hard and soft thresholding. Even though CCSC does not achieve the same degree of sparsity as IST and IHT, the filters learned are seemingly better for the reconstruction task based on their higher PSNR values achieved. Current work focuses on investigating the discriminative power of the proposed representation in classification problems.

REFERENCES

[1] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[2] A. Coates and A. Y. Ng, "The importance of encoding versus training with sparse coding and vector quantization," 2011.
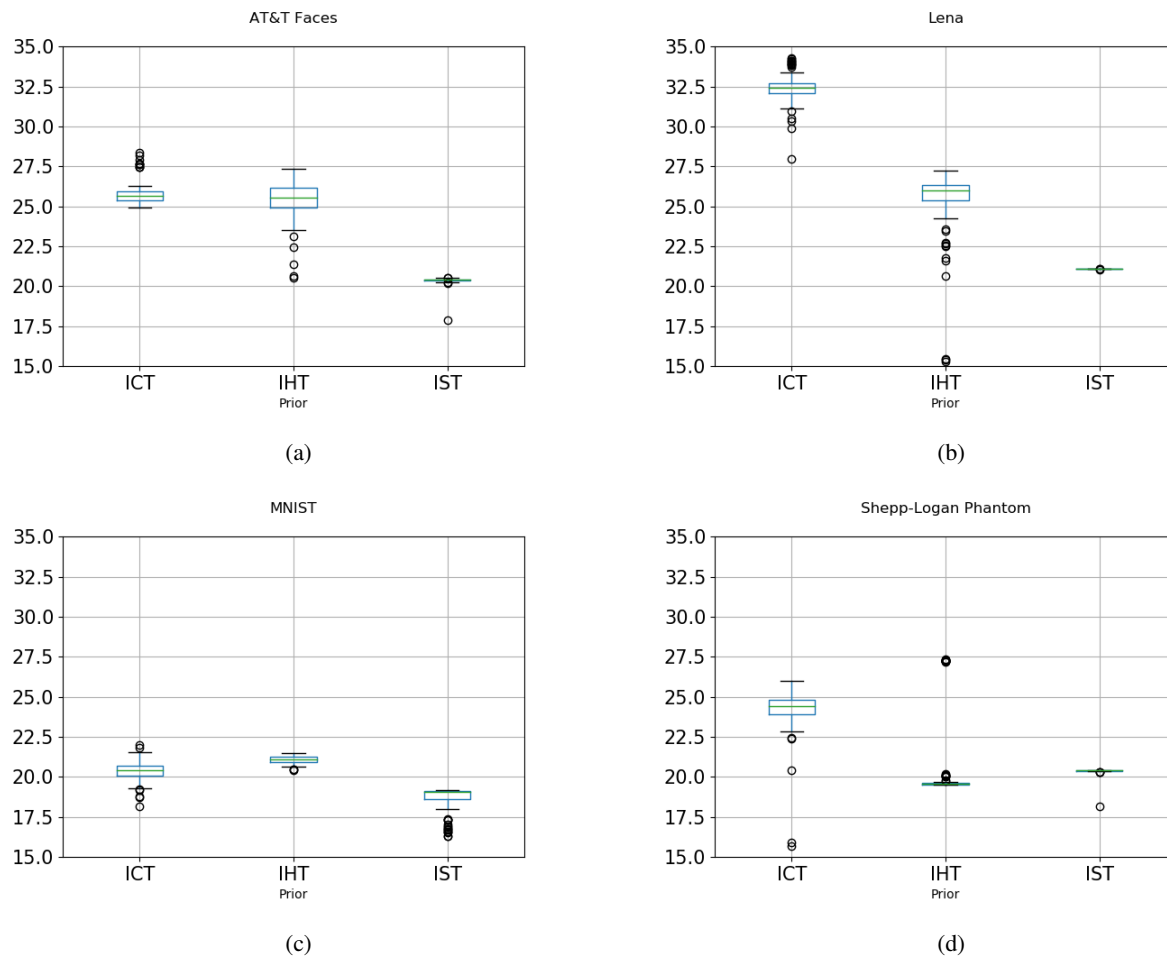
Fig. 8. Boxplots of PSNR for CSC using different algorithms for z-step.

[3] O. Bryt and M. Elad, "Compression of facial images using the k-svd algorithm," *Journal of Visual Communication and Image Representation*, vol. 19, no. 4, pp. 270–282, 2008.

[4] I. Horev, O. Bryt, and R. Rubinstein, "Adaptive image compression using sparse dictionaries," in *2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*. IEEE, 2012, pp. 592–595.

[5] W. Fu, S. Li, L. Fang, and J. A. Benediktsson, "Adaptive spectral–spatial compression of hyperspectral image with sparse representation," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, pp. 671–682, 2016.

[6] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[7] A. Achim, A. Bezerianos, and P. Tsakalides, "Novel bayesian multiscale method for speckle removal in medical ultrasound images," *IEEE transactions on medical imaging*, vol. 20, no. 8, pp. 772–783, 2001.

[8] C. Jiang, Q. Zhang, R. Fan, and Z. Hu, "Super-resolution ct image reconstruction based on dictionary learning and sparse representation," *Scientific reports*, vol. 8, no. 1, pp. 1–10, 2018.

[9] Y. Huang, L. Shao, and A. F. Frangi, "Simultaneous super-resolution and cross-modality synthesis of 3d medical images using weakly-supervised joint convolutional sparse coding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6070–6079.

[10] T. Zhang, B. Ghanem, S. Liu, C. Xu, and N. Ahuja, "Low-rank sparse coding for image classification," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[11] B. Chen, J. Li, B. Ma, and G. Wei, "Convolutional sparse coding classification model for image classification," in *2016 IEEE international conference on image processing (ICIP)*. IEEE, 2016, pp. 1918–1922.

[12] A. Adler, M. Elad, Y. Hel-Or, and E. Rivlin, "Sparse coding with anomaly detection," *Journal of Signal Processing Systems*, vol. 79, no. 2, pp. 179–188, 2015.
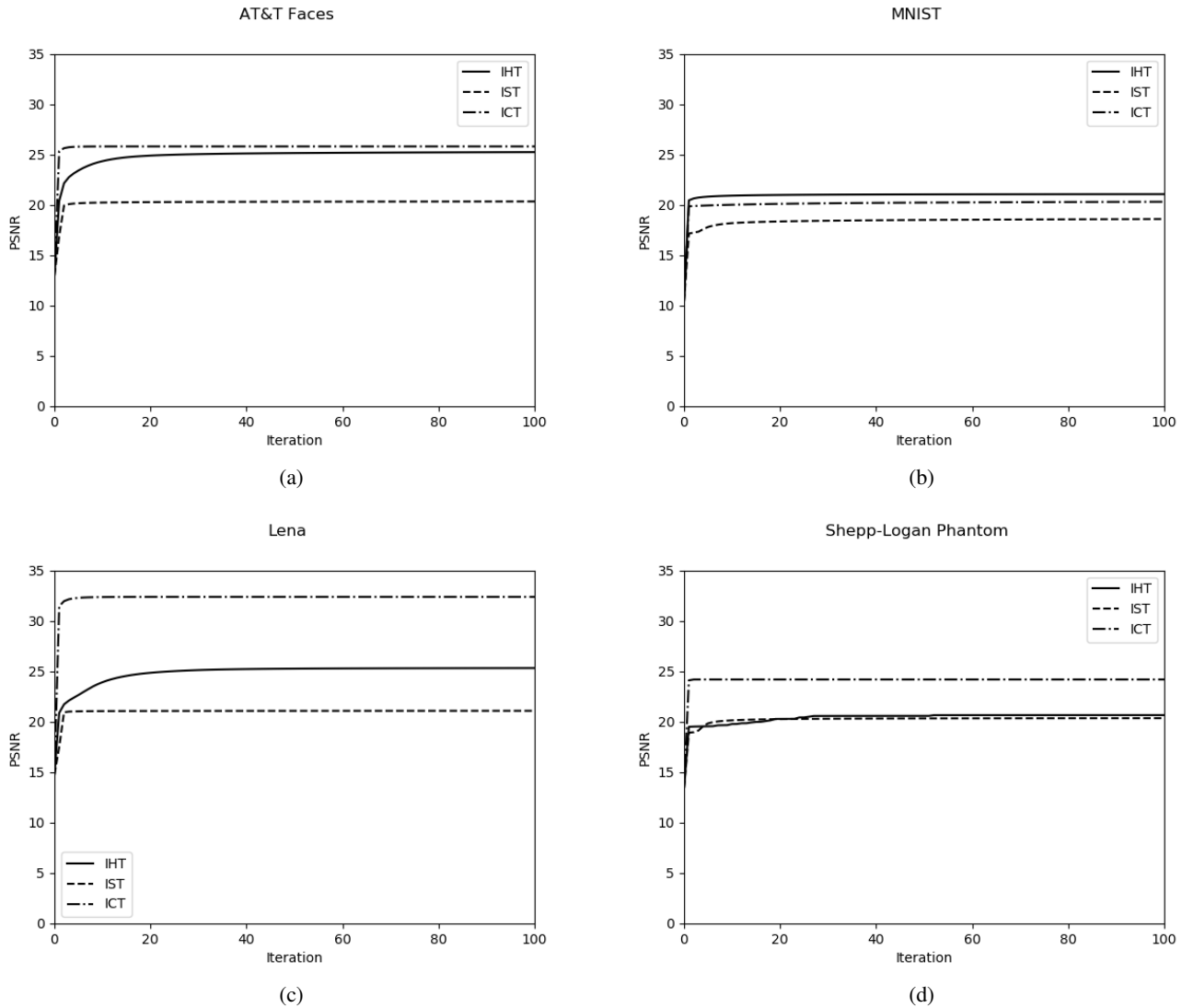
Fig. 9. Learning iterations vs PSNR for CSC using different algorithms for z-step.

[13] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and intelligent laboratory systems*, vol. 2, no. 1-3, pp. 37–52, 1987.

[14] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," *Neural computation*, vol. 7, no. 6, pp. 1129–1159, 1995.

[15] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE journal*, vol. 37, no. 2, pp. 233–243, 1991.

[16] I. Tosic and P. Frossard, "Dictionary learning," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, 2011.

[17] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?" *Vision research*, vol. 37, no. 23, pp. 3311–3325, 1997.

[18] I. Daubechies, M. Defrise, and C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, vol. 57, no. 11, pp. 1413–1457, 2004.

[19] T. Blumensath and M. E. Davies, "Iterative thresholding for sparse approximations," *Journal of Fourier analysis and Applications*, vol. 14, no. 5-6, pp. 629–654, 2008.

[20] D. Malioutov and A. Aravkin, "Iterative log thresholding," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 7198–7202.

[21] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *2010 IEEE Computer Society Conference on*

computer vision and pattern recognition. IEEE, 2010, pp. 2528–2535.

[22] R. Grosse, R. Raina, H. Kwong, and A. Y. Ng, "Shift-invariance sparse coding for audio classification," *arXiv preprint arXiv:1206.5241*, 2012.

[23] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[24] P. Pad, F. Salehi, E. Celis, P. Thiran, and M. Unser, "Dictionary learning based on sparse distribution tomography," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70.* JMLR. org, 2017, pp. 2731–2740.

[25] F. Wen, L. Chu, P. Liu, and R. C. Qiu, "A survey on nonconvex regularization-based sparse and low-rank recovery in signal processing, statistics, and machine learning," *IEEE Access*, vol. 6, pp. 69 883–69 906, 2018.

[26] J.-J. Mei, Y. Dong, T.-Z. Huang, and W. Yin, "Cauchy noise removal by nonconvex admm with convergence guarantees," *Journal of Scientific Computing*, vol. 74, no. 2, pp. 743–766, 2018.

[27] F. Sciacchitano, Y. Dong, and T. Zeng, "Variational approach for restoring blurred images with Cauchy noise," *SIAM Journal on Imaging Sciences*, vol. 8, no. 3, pp. 1894–1922, 2015.

[28] T. Wan, N. Canagarajah, and A. Achim, "Segmentation of noisy colour images using cauchy distribution in the complex wavelet domain," *IET Image Processing*, vol. 5, no. 2, pp. 159–170, 2011.

[29] A. Achim and E. E. Kuruoglu, "Image denoising using bivariate α-stable distributions in the complex wavelet domain," *IEEE Signal Processing Letters*, vol. 12, no. 1, pp. 17–20, 2005.

[30] M. I. H. Bhuiyan, M. O. Ahmad, and M. Swamy, "Spatially adaptive wavelet-based method using the Cauchy prior for denoising the SARimages," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 4, pp. 500–507, 2007.

[31] J. J. Ranjani and S. Thiruvengadam, "Dual-tree complex wavelet transform based SAR despeckling using interscale dependence," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 6, pp. 2723–2731, 2010.

[32] Q. Gao, Y. Lu, D. Sun, Z.-L. Sun, and D. Zhang, "Directionlet-based denoising of SAR images using a Cauchy model," *Signal processing*, vol. 93, no. 5, pp. 1056–1063, 2013.

[33] O. Karakuş, P. Mayo, and A. Achim, "Convergence guarantees for non-convex optimisation with cauchy-based penalties," *arXiv preprint arXiv:2003.04798*, 2020.