SPARSE REPRESENTATION AND DENOISING FOR IMAGES AFFECTED BY GENERALIZED GAUSSIAN NOISE

Florin Ilarion Miertoiu[1] and Bogdan Dumitrescu[2]

In this paper, the image denoising problem is considered, when the image is perturbed with noise generated from a Generalized Gaussian distribution with an unknown shape parameter. We start by extending the algorithm for estimating the shape parameter presented in [1]*, in the context of sparse representations, with an alternative variant for its update step. Both variants of the update step, new and old, combined with the Feasibility Pump adaptation for Generalized Gaussian noise, give better results than ℓ_p versions of Orthogonal Matching Pursuit (OMP) or ℓ_1 regularization (LP_L1). The quality of the sparse representations is comparable with that given by algorithms informed of the true shape parameter value. As an application, we employ our algorithms to the image denoising problem and obtain a favorable comparison, in terms of Peak SNR and SSIM, with other state of the art algorithms using sparse representations.*

Keywords: sparse representations; image denoising; feasibility pump; Generalized Gaussian noise; shape parameter estimation; regularization; probability density function;

1. Introduction

1.1. Problem Formulation

Sparse representations are one of the most important linear optimization problems. They originated from the desire to increase the flexibility of linear representations, and they were successfully used in machine learning, data analysis, compressed sensing, image processing and many other topics. The sparse representation $\boldsymbol{x} \in \mathbb{R}^n$ of a signal $\boldsymbol{y}$ is obtained by solving the linear system $\boldsymbol{y} = \boldsymbol{D}\boldsymbol{x}$, where $\boldsymbol{D} \in \mathbb{R}^{m \times n}$, $m < n$, is the dictionary, under the constraint that most of the coefficients of $\boldsymbol{x}$ are zero. In practice, the signal $\boldsymbol{y}$ is affected by noise, and diverse optimization models were proposed depending on the type of noise.

[1]Department of Automatic Control and Computers, University Politehnica of Bucharest, Romania, e-mail: miertoiu.florin21@gmail.com

[2]Department of Automatic Control and Computers, University Politehnica of Bucharest, Romania, e-mail: bogdan.dumitrescu@upb.ro

A Generalized Gaussian noise is obtained by sampling a Generalized Gaussian distribution [2]. This distribution is used in image processing applications, communication channel modeling, radar, finance and noise modeling. In [3] the problem of communication over an additive noise channel, where the noise is obtained by sampling a Generalized Gaussian distribution, is analyzed. In [4], the Generalized Gaussian distribution is modeled for edge modeling of images. Communication channels affected by Generalized Gaussian noise are also analyzed in [5] for amplitude modulation.

The Generalized Gaussian probability density function is defined as [6]

$$g(\xi;\mu,\sigma,p) = \frac{1}{2\Gamma(1+1/p)A(p,\sigma)} e^{-|\frac{\xi-\mu}{A(p,\sigma)}|^p}, \xi \in \mathbb{R} \tag{1}$$

where μ represents the mean, σ^2 is the variance, p is the shape parameter and $A(p,\sigma)$ is a scaling factor; Γ is the Gamma function. The parameter p dictates the general shape of the distribution; for $p = 1$, the distribution is Laplacian and for $p = 2$, the Gaussian distribution is obtained.

Problems with signals affected by Generalized Gaussian noise lead to optimization programs in which the l_p norm is used. The general form of such a sparse representation problem is

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad & \|\boldsymbol{y} - \boldsymbol{D}\boldsymbol{x}\|_p \\ \text{subject to} \quad & \|\boldsymbol{x}\|_0 \leq K \end{aligned} \tag{2}$$

where K is the sparsity level, namely the number of atoms K from dictionary $\boldsymbol{D}$ that can be used for the representation $\boldsymbol{x}$.

In most applications, problem (2) can be relaxed by replacing the l_0 norm with the l_1 one, which results in the LASSO problem

$$\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimize}} \quad \|\boldsymbol{y} - \boldsymbol{D}\boldsymbol{x}\|_p + \lambda\|\boldsymbol{x}\|_1 \tag{3}$$

1.2. Previous Work

There are only few algorithms for solving problems (2) or (3). OMP-p [7] and LP_L1 [8] assume that the value of p is known. In paper [1], as in this paper, the value of p is not known, and so the focus is both on finding the shape estimation parameter p, with $p \geq 1$, as well as recovering the sparse signal with the smallest representation error.

There are other works on shape parameter estimation, but proposing an a posteriori estimation and not for a sparse model. In [6], an estimator for the shape parameter, using the method of moments, for a Generalized Gaussian Distribution is presented. The method of moment estimation is also used in [9], together with maximum likelihood method, to obtain the shape parameter of the Generalized Gaussian Distribution. Both methods are analyzed separately also in [10]. Other variants of moment-based estimators are proposed in [11].

By matching the entropy of a generalized Gaussian distribution with the empirical data obtained from that distribution, in [12] a new method for obtaining the shape parameter is presented.

Image denoising is one of the important problems in signal processing. Images can be affected by low quality sensors or natural perturbations. Many sparse reconstruction methods showed good results in solving this problem because they can effectively represent the true image while ignoring the noise.

In [13], Bayesian reconstruction is used on a dictionary trained with the K-SVD algorithm in order to perform image denoising on a signal perturbed by Generalized Gaussian Noise. In [14], wavelet decomposition is combined with K-SVD for dictionary training to denoise images affected by Gaussian Noise. In [15], three weight matrices are added to the data and regularization terms in the model for images perturbed by Generalized Gaussian Noise, in order to improve the statistical characterization of the image data. The denoising step is done using the alternating directions method of multipliers (ADM) algorithm.

The Branch and Cut algorithm is adapted for image denoising in [16], using K-SVD for the dictionary learning. Deep Neural Networks can also be used both for the dictionary learning step and the sparsity recovering step in [17]. In [18] two algorithms are proposed for hyperspectral image restoration when the noise that affects the image is Gaussian or Poissonian.

1.3. Contributions

In this paper, the image denoising problem is posed as follows. The original (clean) image $C \in \mathbb{R}^{t\times t}$ is additively perturbed by Generalized Gaussian noise $\boldsymbol{V} \in \mathbb{R}^{t\times t}$ with zero mean and shape parameter p. We are in possession of the noisy image

$$\widetilde{\boldsymbol{C}} = \boldsymbol{C} + \boldsymbol{V} \tag{4}$$

The goal is to recover $\boldsymbol{C}$ as well as possible. To this purpose, we use the following approach. Firstly, using a small number of patches $\boldsymbol{P} \in \mathbb{R}^{s\times s}$ from the perturbed image $\widetilde{\boldsymbol{C}}$, with $s \leq t$, a dictionary $\boldsymbol{D}$ is trained using a dictionary learning algorithm. Then, a matrix $\widetilde{\boldsymbol{Y}} \in \mathbb{R}^{m\times q}$ is built, having as columns vectorized patches $\boldsymbol{P}$ from $\widetilde{\boldsymbol{C}}$. Ideally, all possible distinct patches are used, but practically it may be more appropriate to use only a limited degree of overlap between patches; we denote q the number of patches and $m = s^2$ is the number of pixels in a patch. We represent the patches using the dictionary $\boldsymbol{D}$, by solving the sparse representation problem

$$\begin{aligned} \underset{\boldsymbol{X}\in\mathbb{R}^{n\times q}}{\text{minimize}} \quad & \|\widetilde{\boldsymbol{Y}} - \boldsymbol{D}\boldsymbol{X}\|_p \\ \text{subject to} \quad & \|\boldsymbol{X}_i\|_0 \leq K, i \in \{1, ..., q\} \end{aligned} \tag{5}$$

Problem (5) can be reformulated in LASSO form, which gives the optimization problem

$$\underset{\boldsymbol{X}\in\mathbb{R}^{n\times q}}{\text{minimize}} \quad \|\widetilde{\boldsymbol{Y}} - \boldsymbol{D}\boldsymbol{X}\|_p + \lambda\|\boldsymbol{X}\|_1 \tag{6}$$

Once (5) or (6) are solved, the cleaned image $\hat{\boldsymbol{C}}$ is obtained by averaging the values that a pixel has in the reconstructed patches $\boldsymbol{D}\boldsymbol{X}$ to which it belongs.

While the above denoising scheme has a standard structure, our contribution consists in adapting the solution to Gaussian Generalized noise whose shape parameter p is unknown. We use the algorithms from [1], proposing also a new method for shape parameter estimation. The results that we have obtained with such an approach are comparable to or even better than those obtained with sparse representation algorithms knowing the true value of the shape parameter.

2. Algorithm

2.1. Algorithm for shape parameter estimation

A sparse representation algorithm (SRA) assumes that the noise has a certain distribution. In the framework adopted in this paper, we are interested in Flexible SRAs (FSRA), which can use any given p, like OMP-p [7] and LP_L1 [8].

In most engineering problems the shape parameter p_{true} is not known a priori. In particular, noise affecting images has often an impulsive component, better modeled with a shape parameter $p < 2$, which give distribution with longer tail than the Gaussian.

The shape parameter estimation (SPE) algorithm from [6] uses the error $\widetilde{\boldsymbol{Y}} - \boldsymbol{D}\boldsymbol{X}$ obtained by the FSRA in order to estimate the shape parameter $\tilde{p}$, which is clearly different from p_{true} as well as from the p we have used.

A contribution of this paper is to extend the algorithm for the estimation of p_{true} using a FSRA, presented in [1], with a new update step for the shape parameter. Algorithm 1 shows the operations of our enhanced algorithm.

Both variants of the algorithm start with a given p (we start from 2, as in most cases the noise is assumed to be Gaussian). At each step of our iterative algorithm, the error $\widetilde{\boldsymbol{Y}} - \boldsymbol{D}\boldsymbol{X}$ is computed and the algorithm presented in [6] is used to compute the associated $\tilde{p}$.

The difference between the two variants of the algorithm is given by the updates of the shape parameter p. For the first version of the algorithm, named Half-Adapt, which was proposed in [1], the norm is updated via

$$p \leftarrow (p + \tilde{p})/2. \tag{7}$$

Data: Signals to represent $\widetilde{\boldsymbol{Y}} \in \mathbb{R}^{m \times q}$, dictionary $\boldsymbol{D} \in \mathbb{R}^{m \times n}$, sparsity level $K \in \mathbb{Z}$, maximum number of iterations for norm estimation $Iter_{norm}$, stopping threshold θ
Result: Sparse representations $\boldsymbol{X} \in \mathbb{R}^{n \times q}$, estimated shape parameter $p \in \mathbb{R}$, $p \geq 1$

Compute sparse representations $\boldsymbol{X}$ using algorithm with $p = 2$
Use algorithm in [6] to estimate shape parameter $\tilde{p}$ from representation errors
Update norm p using (7) or (8)
while *number of iterations* $\leq Iter_{norm}$ *or* $|p - \tilde{p}| > \theta$ **do**
    Compute sparse representations $\boldsymbol{X}$ using algorithm with p from the previous step
    Use algorithm in [6] to estimate shape parameter $\tilde{p}$
    Update p using (7) or (8)
end
If update (8) was used and the number of iterations was larger than 1, compute final p as the mean value of all previous values of $\tilde{p}$.
Compute sparse representations $\boldsymbol{X}$ using algorithm with the obtained p.

Algorithm 1: General form of the algorithm for shape parameter estimation

So, we go towards p_{true} as guided by the empirical noise distribution, but temper the change in p in order to prevent oscillations. The sparse representation is computed with the new p and so on.

For the second variant of the algorithm, called Mean-Norm, the norm is updated via

$$p \leftarrow \tilde{p}. \tag{8}$$

For this second algorithm, at the end, an additional step is added where the shape parameter is computed as the mean of all values of p that were found during each iteration using the algorithm from [6]. After this final p is computed, the FSRA is ran one more time with p, in order to get the most appropriate representation.

The idea of the second variant of the algorithm stems from the following practical observations; after the FSRA is ran with $p = 2$, the SRE algorithm from [6] tends to estimate $\tilde{p}$ close to the true value of p; then, as Algorithm 1 runs using the update (8), the value of $\tilde{p}$ further decreases until the value of the estimated p is much smaller than the true p. Computing the mean of the obtained values of p during the algorithm run is used to compensate for this decrease.

These frameworks are designed so that any FSRA can be used with it. The algorithm starts by running the FSRA, for $p = 2$. Using the SPE algorithm, $\tilde{p}$ is computed. The new p is computed using one of the two updates

(7) or (8). At each iteration, the FSRA is evaluated with the new value of p, etc., and the cycle is repeated until the difference $|p - \tilde{p}|$ is under a certain threshold θ or a certain number of iterations $Iter_{norm}$ was completed. If the Mean-Norm update (8) is used, after the main loop is completed, the mean of all obtained values of p is computed and the FSRA is run for the obtained value. This is the final value of p.

The idea of both variants is to compensate sufficiently for the drop of $\tilde{p}$, computed by the SPE algorithm in [6]. By just taking the value obtained by the algorithm and simply replacing it, the norm usually will go to a smaller value than the original one or in some cases close to 1. The Mean-Norm variant tries to compensate for this by taking the mean of all values obtained, while the Half-Adapt framework slows the drop by taking the new value of the shape parameter as the mean between the previous shape parameter and the obtained parameter by the SRE.

2.2. Image Denoising using the shape parameter estimator

The image denoising framework derived from Algorithm 1 is shown in Algorithm 2. A dictionary learning algorithm, working with the p_d norm of the error, is used to obtain the dictionary $\boldsymbol{D}$; here, p_d is a rough estimation of the true shape parameter. Each patch of the noisy image is represented using the trained dictionary $\boldsymbol{D}$ and Algorithm 1, where we use OMP-p as FSRA. Using the representations $\boldsymbol{X}$, the cleaned patches are reconstructed as $\boldsymbol{Y}_{rec} = \boldsymbol{DX}$. The image is reconstructed by using these patches. Each pixel of the image is computed as the mean of the pixel values in all the patches containing that pixel.

Data: Set of patches to be represented $\widetilde{\boldsymbol{Y}} \in \mathbb{R}^{m \times q}$, shape parameter for dictionary learning p_d, sparsity level $K \in \mathbb{Z}$, maximum number of iterations for norm estimation $Iter_{norm}$, stopping threshold θ
Result: Denoised image
1 Train the dictionary $\boldsymbol{D}$ using any dictionary learning algorithm that optimizes using the p_d norm of the error
2 Compute sparse representations $\boldsymbol{X}$ and estimated p using Algorithm 1
3 For each pixel of the image, compute the mean of values of that pixel in the patches $\boldsymbol{DX}$ that contain it

Algorithm 2: Algorithm for image denoising

3. Results

3.1. Shape parameter estimation for artificial data

In order to test the two versions of shape parameter update in Algorithm (1), dictionaries of size 50×100 and signals $\boldsymbol{y}$ with sparsity level $K \in \{5, 7, 9\}$

are generated. The additive perturbation noise is Generalized Gaussian and its variance is chosen such that the signal to noise ratio is 30. The shape parameters used for the noise are $p_{\text{true}} \in \{1.2, 1.4, 1.6, 1.8\}$.

The results obtained by the variants are the sparse representations $\boldsymbol{X}$ and the shape parameter p.

The sparse representation algorithms that are integrated into Algorithm 1 are FP-GGN [1], OMP-p [7] and LP_L1 [8]. FP-GGN is the adaptation of the Feasibility Pump for the case of a signal perturbed with Generalized Gaussian noise.

The algorithms are tested on a machine with a 6-core processor and 32 GB of RAM and compared in terms of mean representation errors, recovery errors and estimated shape parameters.

The training step in Algorithm 2 is done using $p_d = p_{\text{true}}$ in step 1.

The representation error is computed using the l_p norm with

$$e = \frac{\|\boldsymbol{D}\boldsymbol{x} - \boldsymbol{y}\|_p}{\|\boldsymbol{y}\|_p}, \tag{9}$$

where $\boldsymbol{x}$ is the computed solution, in accordance with the formulation of the initial problem (2). The relative recovery error is computed using

$$\boldsymbol{e}_{rec} = \frac{\|\boldsymbol{x} - \boldsymbol{x}_{\text{true}}\|_p}{\|\boldsymbol{y}\|_p}, \tag{10}$$

where $\boldsymbol{x}_{\text{true}}$ is the true representation of the signal $\boldsymbol{y}$ using the dictionary $\boldsymbol{D}$ in the unperturbed case.

In Table 1, the value of the estimated shape parameter is shown for the Half-Adapt and Mean-Norm versions of Algorithm 1. For each value of K and p, the estimated p that is closest to the true value p_{true} is written in bold. It can be seen that Mean-Norm offers the closest approximation to the true value of p in most cases. For $K = 5$ and $K = 7$, FP-GGN gives the closest approximation. As K increases, LP_L1 appears to have the closer approximation. We note that all algorithms give estimated values that are near p_{true}.

In Table 2, the mean representation errors are displayed. HA denotes the Half-Adapt version of the algorithm, MN denotes the Mean-Norm and F denotes the version with the Fixed value of the norm. For each K, there are three columns: the left and center ones contain the errors of the adaptive algorithms for which p_{true} is unknown (the two versions of Algorithm 1); the right one contains the errors of the same basic algorithms that are in possession of p_{true}. Similarly to Table 1, for each value of K and p, the smallest recovery error is written in bold. FP-GGN and LP_L1 have similar values of the representation errors, with very small differences. As K increases, FP-GGN begins to have a slightly smaller representation error when compared to LP_L1.

In Table 3, the mean recovery errors are displayed. It can be seen in multiple cases that the mean representation error is better for FP-GGN. Here,

Table 1

Mean Estimated Shape Parameters for OMP-p (top in each cell), FP-GGN (middle) and LP_L1 (bottom)

K	5		7		9	
	Half-Adapt	Mean-Norm	Half-Adapt	Mean-Norm	Half-Adapt	Mean-Norm
p=1.2	1.134	**1.212**	1.088	1.190	1.067	1.235
	1.148	1.215	1.107	**1.199**	1.051	1.175
	1.187	1.297	1.133	1.261	1.064	**1.193**
p=1.4	1.326	1.386	1.232	1.348	1.176	1.303
	1.348	**1.399**	1.260	**1.367**	1.187	1.322
	1.365	1.427	1.354	1.446	1.240	**1.427**
p=1.6	1.535	1.587	1.448	1.541	1.284	1.440
	1.559	**1.608**	1.502	**1.579**	1.337	**1.483**
	1.577	1.640	1.561	1.690	1.432	1.463
p=1.8	**1.811**	1.830	1.698	1.755	1.675	1.727
	1.813	1.833	1.707	1.763	1.676	**1.749**
	1.883	1.893	1.750	**1.831**	1.661	1.680

Table 2

Mean Representation Errors for OMP-p (top in each cell), FP-GGN (middle) and LP_L1 (bottom)

K	5			7			9		
	HA	MN	F	HA	MN	F	HA	MN	F
p=1.2	0.035	0.035	0.032	0.037	0.037	0.036	0.064	0.063	0.058
	0.028	**0.028**	**0.028**	**0.027**	0.028	0.028	**0.027**	**0.027**	0.028
	0.028	0.029	**0.028**	**0.027**	0.029	0.028	**0.027**	**0.027**	**0.027**
p=1.4	**0.029**	**0.029**	**0.029**	0.041	0.041	0.038	0.037	0.036	0.034
	0.029	**0.029**	**0.029**	0.029	0.029	0.029	0.028	0.028	0.028
	0.029	**0.029**	**0.029**	0.029	**0.028**	0.029	0.028	**0.027**	0.029
p=1.6	**0.029**	**0.029**	**0.029**	0.031	0.031	0.030	0.042	0.042	0.039
	0.030	0.030	0.030	**0.029**	**0.029**	**0.029**	**0.028**	**0.028**	**0.028**
	0.030	0.030	0.030	**0.029**	0.030	**0.029**	0.029	0.029	0.029
p=1.8	0.031	0.031	0.031	0.033	0.033	0.034	0.038	0.040	0.043
	0.030	**0.030**	**0.030**	**0.029**	**0.029**	**0.029**	**0.028**	**0.028**	**0.028**
	0.030	**0.030**	**0.030**	**0.029**	**0.029**	**0.029**	0.029	0.029	**0.028**

as well as in Table 2, the adaptive algorithms give values that are very near to the algorithms knowing p_{true}. This shows that both versions of our Algorithm 1 are able to provide good sparse representations without the knowledge of the shape parameter value.

Regarding support recovery, in average, FP-GGN Half-Adapt has 0.233 false negatives per test (true support recovered in 78.3% of cases, OMP-p Half-Adapt has 0.750 false negatives per test (true support recovered in 64.1% of cases) and LP_L1 Half-Adapt has 0.341 false negatives per test (true support recovered in 73.3% of cases). It can be seen that FP-GGN gets the correct support in more cases that the other two algorithms.

For the Mean-Norm framework, in average, FP-GGN has 0.241 false negatives per test (true support recovered in 77.5% of cases), OMP-p has 0.667 false negatives per test (true support recovered in 65% of cases) and LP_L1 has 0.244 false negatives per test (true support recovered in 72.5% of cases). Similarly with the previous framework FP-GGN outperforms both algorithms.

Table 3

Mean Recovery Errors for OMP-p (top in each cell), FP-GGN (middle) and LP_L1 (bottom)

K	5			7			9		
	HA	MN	F	HA	MN	F	HA	MN	F
p=1.2	0.024	0.024	0.017	0.030	0.029	0.026	0.122	0.112	0.093
	0.010	**0.010**	**0.010**	**0.013**	**0.013**	**0.013**	0.016	**0.015**	**0.015**
	0.010	**0.010**	**0.010**	**0.013**	**0.013**	**0.013**	0.016	0.016	**0.015**
p=1.4	**0.010**	**0.010**	**0.010**	0.037	0.037	0.032	0.043	0.040	0.033
	0.011	0.011	0.011	0.015	**0.014**	**0.014**	0.018	**0.017**	0.018
	0.013	0.011	0.013	0.016	0.015	0.015	0.020	0.020	0.019
p=1.6	**0.011**	**0.011**	**0.011**	0.016	0.016	0.015	0.044	0.044	0.039
	0.012	0.014	0.014	**0.013**	**0.013**	**0.013**	0.017	0.044	**0.016**
	0.016	0.012	0.012	0.014	0.014	0.014	0.019	0.017	0.018
p=1.8	0.016	0.016	0.014	0.020	0.020	0.022	0.041	0.047	0.047
	0.012	**0.012**	**0.012**	**0.014**	**0.014**	**0.014**	**0.018**	0.047	**0.018**
	0.012	0.013	**0.012**	0.015	0.015	**0.014**	0.019	**0.018**	**0.018**

For the Half-Norm framework, the mean running time of FP-GGN is 4056s per test, OMP-p takes 74.3s and LP_L1 takes 11.8s. The Mean-Norm framework is faster for the FP-GGN, which needs, in average, 3257s, and for OMP-p which needs 53.2s. The Mean-Norm framework is slower when the LP_L1 algorithms is used and needs 14.6s.

It can be seen that the Mean-Norm framework improves the running time by around 20% for the FP-GGN and OMP-p algorithms when compared to the adapt framework, but at the cost of a small loss of the support recovery of the algorithms.

3.2. Image Denoising using the shape parameter estimator

For the image denoising tests, Algorithm 1 is used with OMP-p and is compared with OMP, which optimizes using l_2, OMP-p, which knows the value of p_{true} and the Orthogonal Wavelet Thresholding (OWT) algorithm presented in [19], which uses wavelets for image denoising. The focus is to find out which algorithm can recover the shape parameter of the noise distribution better and which one gives a larger peak SNR (PSNR) and a better structural similarity index (SSIM). The PSNR is computed with the MATLAB command *psnr* and the SSIM is evaluated using the MATLAB command *ssim*.

The dictionary $\boldsymbol{D}$ is trained using 2560 patches from the perturbed image, using a dictionary learning procedure; we take $p_d = p_{\text{true}}$ in Algorithm 2. Another larger set of 15876 patches is then extracted and is used for the actual process of image denoising. The perturbed images used for the test have a mean $PSNR = 25.06$.

The dictionary size is 64×128. The patches are of size 8×8. The sparsity level is $K = 8$. The shape parameter values are $p \in \{1.2, 1.4, 1.6, 1.8\}$. The value of the SNR is 20. Tests are done on three well known images: Lena, Barbara and Boat. From each image a large patch of 256×256 is selected from which the training and learning patches are selected. Each image is perturbed

Table 4

Mean Estimated Shape Parameters for Mean-Norm and Half-Adapt

	Mean-Norm	Half-Adapt
p=1.2	1.498	**1.154**
p=1.4	**1.576**	1.177
p=1.6	**1.708**	1.417
p=1.8	**2**	**2**

Table 5

Mean PSNR for OMP, OMP-p, Mean-Norm, Half-Adapt and OWT

	OMP	OMP-p	Mean-Norm	Half-Adapt	OWT
p=1.2	29.629	29.882	29.791	**29.897**	29.76
p=1.4	29.593	29.731	29.703	**29.765**	29.74
p=1.6	29.620	29.640	29.640	29.687	**29.74**
p=1.8	**29.737**	29.733	**29.737**	**29.737**	29.67

with Generalized Gaussian noise such as the desired SNR is obtained. For each combination of image and value of p, 5 tests are done.

Table 4 shows the mean shape parameter estimation for the Mean-Norm and Half-Adapt versions of Algorithm 2. In bold, the estimation that is closest to the real value is written. It can be seen that the Half-Adapt version tends to underestimate the shape parameter, while the Mean-Norm version tends to overestimate. It can be seen that for $p = 1.2$, the Half-Adapt estimation of the parameter is closest to the real value of p, while for $p = 1.4$ and $p = 1.6$, Mean-Norm estimates closer to the real value, in many tests the value being estimated at 2. For $p = 1.8$, both versions estimate the shape parameter at 2.

In Table 5, the mean PSNR values are presented for OMP, OMP-p, Mean-Norm, Half-Adapt and OWT algorithms. For each value of p, the largest PSNR values are written in bold. Mean-Norm has a better peak SNR than OMP, while having a smaller value than OMP-p. Half-Adapt has the best value in all cases, except for $p = 1.6$. For $p = 1.8$, Mean-Norm and Half-Adapt have the same value as OMP, as they estimate the shape parameter is estimated with the value of 2.

The mean values of the SSIM are presented in Table 6, with the best value being written in bold. For all cases, it can be seen that OWT gives the best value of the SSIM; OWT, despite its lower PSNR performance, is more adequate to images. Mean-Norm is better than OMP, but has a smaller value than the other algorithms. Half-Adapt proves to have once again the best result in the class of tested sparse representation algorithms. For $p = 1.8$, all algorithms, except OWT, have the same result, as both variants of Algorithm 2 estimate the shape parameter to 2.

Table 6

Mean Structural Similarity Index for OMP, OMP-p, Mean-Norm, Half-Adapt and OWT

	OMP	OMP-p	Mean-Norm	Half-Adapt	OWT
p=1.2	0.794	0.812	0.804	0.814	**0.853**
p=1.4	0.795	0.804	0.801	0.809	**0.856**
p=1.6	0.795	0.799	0.798	0.802	**0.858**
p=1.8	0.800	0.800	0.800	0.800	**0.860**

In average, for 15876 patches, the running time is 5097 seconds for Half-Adapt, 3779 seconds for Mean-Norm, 751 seconds for OMP-p with known norm ,0.002 seconds for OMP and 0.087 seconds for OWT. It can be seen that Mean-Norm is faster than Half-Adapt. The running time increases as the decrease of the value of the estimated shape parameter is attenuated.

4. Conclusions

In this paper, an extension for the framework presented in [1] was presented. It can be used with any FSRA. For this general framework, two version with different adaptation steps for the shape parameter p have been considered and implemented, Half-Adapt and Mean-Norm. Both versions can find shape parameters that are close to the true value of p. Also, the mean representation and mean recovery error are similar to the algorithms which use the real value of p. Mean-Norm is faster than Half-Adapt and it also offers better support recovery. Results are especially good at small values of K. Algorithm 1 is also extended to the image denoising problem. Both variants of this algorithm, Half-Adapt and Mean-Norm are tested, using OMP-p as the FSRA, and compared against OMP and OMP-p with p_{true} and OWT. It can be seen, in this case, that Half-Adapt has the best values for the PSNR and for the SSIM, excepting the OWT algorithm in the latter case. Mean-Norm is better than OMP, estimates better the shape parameter p than Half-Adapt and is faster. So, despite their complexity, which can be improved with a dedicated implementation, our proposed algorithm can bring clear benefits to image denoising.

REFERENCES

[1] F. Miertoiu and B. Dumitrescu, "Shape parameter and sparse representation recovery under generalized gaussian noise," in *EUSIPCO*, (Dublin, Ireland), 2021.

[2] A. Dytso, R. Bustin, H. Poor, and S. Shamai, "Analytical properties of generalized gaussian distributions," *Journal of Statistical Distributions and Applications*, vol. 5, no. 1, pp. 1–40, 2018.

[3] A. Dytso, R. Bustin, V. Poor, and S. Shitz, "On additive channels with generalized gaussian noise," in *2017 IEEE International Symposium on Information Theory (ISIT)*, pp. 426–430, IEEE, 2017.

[4] C. Bouman and K. Sauer, "A generalized gaussian image model for edge-preserving map estimation," *IEEE Transactions on image processing*, vol. 2, no. 3, pp. 296–310, 1993.

[5] H. Soury, F. Yilmaz, and M. Alouini, "Error rates of m-pam and m-qam in generalized fading and generalized gaussian noise environments," *IEEE Communications Letters*, vol. 17, no. 10, pp. 1932–1935, 2013.

[6] J. Dominguez-Molina, G. González-Farías, R. Rodríguez-Dagnino, and I. C. Monterrey, "A practical procedure to estimate the shape parameter in the generalized gaussian distribution," 2003. Available at `http://www.cimat.mx/reportes/enlinea/I-01-18_eng.pdf`.

[7] W. Zeng, H. So, and X. Jiang, "Outlier-Robust Greedy Pursuit Algorithms in ℓ_p-Space for Sparse Approximation," *IEEE Trans. Signal Processing*, vol. 64, no. 1, pp. 60–75, 2016.

[8] F. Wen, P. Liu, Y. Liu, R. C. Qiu, and W. Yu, "Robust sparse recovery in impulsive noise via ℓ_p-ℓ_1 optimization," *IEEE Transactions on Signal Processing*, vol. 65, no. 1, pp. 105–118, 2016.

[9] M. K. Varanasi and B. Aazhang, "Parametric generalized gaussian density estimation," *The Journal of the Acoustical Society of America*, vol. 86, no. 4, pp. 1404–1415, 1989.

[10] S. Meignen and H. Meignen, "On the modeling of small sample distributions with generalized gaussian density in a maximum likelihood framework," *IEEE Transactions on Image Processing*, vol. 15, no. 6, pp. 1647–1652, 2006.

[11] Y. Chen and N. C. Beaulieu, "Novel low-complexity estimators for the shape parameter of the generalized gaussian distribution," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 4, pp. 2067–2071, 2008.

[12] B. Aiazzi, L. Alparone, and S. Baronti, "Estimation based on entropy matching for generalized gaussian pdf modeling," *IEEE Signal Processing Letters*, vol. 6, no. 6, pp. 138–140, 1999.

[13] 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.

[14] H. Li and F. Liu, "Image denoising via sparse and redundant representations over learned dictionaries in wavelet domain," in *2009 Fifth International Conference on Image and Graphics*, pp. 754–758, IEEE, 2009.

[15] J. Xu and D. Zhang, L.and Zhang, "A trilateral weighted sparse coding scheme for real-world image denoising," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 20–36, 2018.

[16] Y. Liu, S. Canu, P. Honeine, and S. Ruan, "Mixed integer programming for sparse coding: Application to image denoising," *IEEE Transactions on Computational Imaging*, vol. 5, no. 3, pp. 354–365, 2019.

[17] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Advances in neural information processing systems*, pp. 341–349, 2012.

[18] L. Zhuang and J. M. Bioucas-Dias, "Fast hyperspectral image denoising and inpainting based on low-rank and sparse representations," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 11, no. 3, pp. 730–742, 2018.

[19] F. Luisier and T. Blu, "Sure-let multichannel image denoising: Interscale orthonormal wavelet thresholding," *IEEE Transactions on Image processing*, vol. 17, no. 4, pp. 482–492, 2008.