AUDASCITY: AUdio Denoising by Adaptive Social CosparsITY

Clément Gaultier, Srđan Kitić, Nancy Bertin, Rémi Gribonval

AUDASCITY: AUdio Denoising by Adaptive Social CosparsITY

Clément Gaultier
Inria, Centre Inria Rennes
Rennes, France

Srđan Kitić
Technicolor,
Cesson-Sevigné, France

Nancy Bertin
IRISA - CNRS UMR 6074,
Rennes, France

Rémi Gribonval
Inria, Centre Inria Rennes
Rennes, France

***Abstract*—This work aims at introducing a new algorithm, AUDASCITY, and comparing its performance to the time-frequency block thresholding algorithm for the ill-posed problem of audio denoising. We propose a heuristics which combines time-frequency structure, cosparsity, and an adaptive scheme to denoise audio signals corrupted with white noise. We report that AUDASCITY outperforms state-of-the-art for each numerical comparison. While there is still room for some perceptual improvements, AUDASCITY's usefulness is shown when used as a front-end for a classification task.**

I. Introduction

Denoising is one of the most intensively studied inverse problems in audio signal processing. Whether it originates from the environment or the microphones, noise is an inevitable (and, usually, undesirable) component of audio recordings, calling for a denoising stage in signal processing pipelines for applications such as music transcription, sound classification, speech recognition and many others. To address the noise problem, numerous approaches arose. Some of these use statistical models [1], [2], others use spectral subtraction [3] or thresholding operators [4].

In the last two decades, a body of work addressing reconstruction and inverse problems in audio popularized time-frequency (TF) sparse regularization, from the synthesis or the analysis (cosparse) point of view [5], [6]. While the synthesis approaches comprise a vast majority of these, it has been demonstrated recently [6], [7] that the analysis model can sometimes be advantageous, namely in terms of lower computational cost. Additionally, it was recognized that group sparse models, such as the so-called *social sparsity* [8], may be more adapted to TF structures of audio signals.

The cosparse model alleges that the product $\mathbf{z} \overset{\text{def}}{=} \mathbf{\Omega x}$ is approximately sparse, where vector $\mathbf{x}$ is the target signal and matrix $\mathbf{\Omega}$ is the so-called (time-frequency) *analysis operator*.

Social sparsity aims at approximating the solution of a problem for which the non-zero coefficients of a *sparse* signal $\mathbf{z}$ appear in the form of overlapping groups, with some known structure. However, TF structures vary from one audio signal to another, hence there is a need for a more flexible approach. A recent work based on Persistent Empirical Wiener (PEW) shrinkage [9] allows to recover sparse signals by identifying structures in the TF plane. PEW is used as sparsity enforcing operator akin to the soft-thresholding stage in typical iterative sparse recovery algorithms.

To highlight a given TF structure, algorithms based on social sparsity [10] must explore the entire TF plane, which induces a high computational cost. On the other hand, computationally efficient schemes based on cosparsity, *e.g. A-SPADE* [7], process each segment of an audio sequence independently, thus somehow neglecting its long-term temporal structure. The idea behind this work is to show how coupling the cosparse model and the social shrinkage heuristics together can be beneficial to audio restoration, with audio denoising as a use case.

Thus, in the following, we introduce an algorithm which allows to retain at the same time favorable computational properties of the cosparse prior, and the ability of the social prior to emphasize TF useful structures. It is also able to locally adapt the TF structure it is seeking, instead of choosing one for the entire signal. First, we introduce the algorithm and an accelerated version of it. Then, we present experimental results of two kinds: numerical performance and success indicators on classification task. Finally, we discuss perceptual performance of denoising.

II. AUDASCITY Algorithm

The forward (degradation) model for the case of additive noise is simply the vector sum

$$\widetilde{\mathbf{y}}_{\mathsf{n}} = \widetilde{\mathbf{x}}_{\mathsf{n}} + \widetilde{\mathbf{e}}_{\mathsf{n}}, \tag{1}$$

with $\widetilde{\mathbf{x}}_{\mathsf{n}} \in \mathbb{R}^{\mathsf{L}}$ the n^{th} overlapping frame of a discrete time-domain audio signal $\widetilde{\mathbf{x}}$. $\widetilde{\mathbf{e}}_{\mathsf{n}} \in \mathbb{R}^{\mathsf{L}}$ is modeled as white Gaussian noise of variance σ^2 (assumed given or estimated). From model (1) we window the segment $\widetilde{\mathbf{y}}_{\mathsf{n}}$ such that $\mathbf{y}_{\mathsf{n}} = \mathbf{W}\widetilde{\mathbf{y}}_{\mathsf{n}}$. Here $\mathbf{W} = \text{diag}(\mathbf{w})$ with $\mathbf{w} \in \mathbb{R}^{\mathsf{L}}$ the weighting window. We will denote by $\mathbf{Y}_{\mathsf{n}} \in \mathbb{R}^{\mathsf{L}\times(2\mathsf{b}+1)}$ (resp. $\mathbf{X}_{\mathsf{n}}$) a matrix containing the noisy (resp. clean) frames indexed by $[\mathsf{n}-\mathsf{b}, \mathsf{n}+\mathsf{b}]$. We also define $\mathbf{Z}_{\mathsf{n}} \in \mathbb{C}^{\mathsf{L}\times(2\mathsf{b}+1)}$ the frequency representation of $\mathbf{X}_{\mathsf{n}}$ such that $\mathbf{Z}_{\mathsf{n}} = \mathbf{A}\mathbf{X}_{\mathsf{n}}$. Here $\mathbf{A} \in \mathbb{C}^{\mathsf{P}\times\mathsf{L}}$ is a tight frame, with $\mathsf{P} \geqslant \mathsf{L}$, that performs some local analysis e.g. a possibly zero-padded Fourier transform. Due to the overlap between windows, $\mathbf{Z}_{\mathsf{n}}$ is a local redundant analysis TF representation (Gabor transform) of the underlying audio signal $\widetilde{\mathbf{x}}$.

A. Social Cosparse Modeling

We define time-frequency patterns $\mathit{\Gamma} \in \mathbb{R}^{\mathsf{F}\times\mathsf{T}}$ which embody a TF structure. Examples of $\mathit{\Gamma}$ which we consider here as binary masks are given in Figure (1) (see section III-A for details). Rows account for the frequency dimension and

columns for the time. Fusing the concepts of cosparsity and social sparsity allows us to promote a social sparse prior for the redundant TF transforms $\mathbf{Z}_{\mathsf{n}}$ of the signal $\widetilde{\mathbf{x}}$, hence a socially *cosparse* prior for $\widetilde{\mathbf{x}}$ and the name of the proposed algorithm: "*AUdio Denoising by Adaptive Social CosparsITY (AUDASCITY)*".

B. Social Shrinkage

For clarity in the presentation, in the following we will omit the n index and consider $\hat{\mathbf{X}}, \mathbf{Y}, \mathbf{Z}$, matrices of size $\mathsf{L} \times (2\mathsf{b}+1)$ centered on frame n.

To exploit social cosparsity in the denoising process described next, we use the Persistent Empirical Wiener Shrinkage operator (defined in [9] as thresholding) as a sparsifying step in the inner loop. This shrinkage explicitly includes a time-frequency *pattern* Γ which promotes local TF structures around each TF point. Let $\mathbf{Z} \in \mathbb{C}^{\mathsf{L} \times (2\mathsf{b}+1)}$ be a local TF representation, let ij be coordinates of a TF point in $\mathbf{Z}$ and $\mathbf{P}_{\mathsf{ij}}$ the indexes corresponding to a binary TF patch of size $\mathsf{F} \times \mathsf{T}$ centered in ij. $\mathbf{Z}_{\mathbf{P}_{\mathsf{ij}}} \in \mathbb{C}^{\mathsf{F} \times \mathsf{T}}$ is the matrix extracted from $\mathbf{Z}$ on these indexes. PEW is defined as follows:

$$\mathcal{S}_\mu(\mathbf{Z}|\Gamma)_{(\mathsf{ij})} = \mathbf{Z}_{(\mathsf{ij})} \cdot \left(1 - \frac{\mu^2}{\|\mathbf{Z}_{\mathbf{P}_{\mathsf{ij}}} \circ \Gamma\|_2^2}\right)_+, \qquad (2)$$

Where $(\cdot)_+ = \max(\cdot, 0)$ is the positive part. The shrinkage operator returns a "shrinked" vector (controlled by μ) following a pattern given by Γ. $\circ$ denotes the Hadamard product. Practically, as $\mathcal{S}_\mu(\cdot)_{(\mathsf{ij})}$ is applied component-wise, it can be computed through multidimensional convolution in the Fourier domain.

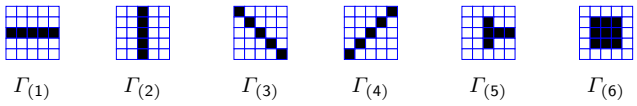


Figure 1. Extended set of time-frequency neighborhoods

C. Overview of the algorithm

As in A-SPADE [7], the technique stands on the *Alternating Direction Method of Multipliers* (ADMM) framework [11]. The denoising procedure is based on two loops. The inner loop performs the denoising given a predefined neighborhood pattern Γ. The outer loop chooses the "optimal" pattern among a subset of possible predefined patterns $\left\{\Gamma_{(\mathsf{k})}\right\}_{\mathsf{k}=1\ldots\mathsf{g}}$ based on the inner loop's estimate $\hat{\mathbf{X}}_{\mathsf{n}}$ obtained with a given Γ.

D. Inner loop

This part of the algorithm denoises $\mathbf{Y}$ given a predefined TF pattern Γ. The pseudo code for this inner loop is given in Algorithm 1 below with, $0 < \alpha < 1$ and $\varepsilon = (2\mathsf{b}+1)\sigma\sqrt{\sum_{\mathsf{j}=1}^{\mathsf{L}} \mathbf{w}_{\mathsf{j}}}$, where $\mathbf{w}_{\mathsf{j}}$ is the j^{th} entry of the window $\mathbf{w}$ and σ^2 is the noise variance.

Note that the $\hat{\mathbf{X}}^{(\mathsf{i})}$ update step admits a closed-form solution, given the tight-frame assumption, similar to [7]. We compactly write the inner loop algorithm as a parameterized procedure: $\hat{\mathbf{X}} = \mathcal{A}(\mathbf{Y}, \Gamma, \alpha, \mu^{(0)}, \varepsilon, \beta, \mathsf{i}_{\max})$.

Algorithm 1 AUDASCITY Inner loop

Require: $\mathbf{A}, \mathbf{Y}, \Gamma, \alpha, \mu^{(0)}, \varepsilon, \beta, \mathsf{i}_{\max}$
  $\hat{\mathbf{X}}^{(0)} = \mathbf{Y}$, $\mathbf{U}^{(0)} = 0$, $\mathsf{i} = 1$
  $\mathbf{Z}^{(\mathsf{i})} = \mathcal{S}_\mu(\mathbf{A}\hat{\mathbf{X}}^{(\mathsf{i}-1)} + \mathbf{U}^{(\mathsf{i}-1)} \mid \Gamma)$
  $\hat{\mathbf{X}}^{(\mathsf{i})} = \underset{\mathbf{X}}{\operatorname{argmin}} \|\mathbf{A}\mathbf{X} - \mathbf{Z}^{(\mathsf{i})} + \mathbf{U}^{(\mathsf{i}-1)}\|_{\mathrm{F}}^2$
    subject to $\|\mathbf{X} - \mathbf{Y}\|_{\mathrm{F}} \leq \varepsilon$
  $\mu^{(\mathsf{i})} = \alpha\mu^{(\mathsf{i}-1)}$
  if $\frac{\|\mathbf{A}\mathbf{X}^{(\mathsf{i})} - \mathbf{Z}^{(\mathsf{i})}\|_{\mathrm{F}}}{\|\mathbf{A}\mathbf{X}^{(\mathsf{i})}\|_{\mathrm{F}}} \leq \beta$ or $\mathsf{i} \geq \mathsf{i}_{\max}$ **then**
    terminate
  else
    $\mathbf{U}^{(\mathsf{i})} = \mathbf{U}^{(\mathsf{i}-1)} + \mathbf{A}\hat{\mathbf{X}}^{(\mathsf{i})} - \mathbf{Z}^{(\mathsf{i})}$
    $\mathsf{i} \leftarrow \mathsf{i} + 1$
  end if
  return $\hat{\mathbf{X}}^{(\mathsf{i})}$

Algorithm 2 AUDASCITY

Require: $\mathbf{Y}, \left\{\Gamma_{(\mathsf{k})}\right\}_{\mathsf{k}=1\ldots\mathsf{g}}, \alpha, \varepsilon, \beta$
  $\mathsf{i}_{\max} = 10$
  for $\mathsf{k} = 1$ to g **do**
    $\mu_{(\mathsf{k})}^{(0)} = \|\Gamma_{(\mathsf{k})}\|_0 \times \|\mathrm{vec}(\mathbf{Y})\|_\infty$
    $\hat{\mathbf{X}}_{(\mathsf{k})} = \mathcal{A}(\mathbf{Y}, \Gamma_{(\mathsf{k})}, \alpha, \mu_{(\mathsf{k})}^{(0)}, \varepsilon, \beta, \mathsf{i}_{\max})$
    Compute $\mathrm{e}_{\Gamma_{(\mathsf{k})}}$ as in (3)
    $\mathsf{m} = \arg\max_{\mathsf{k}} \mathrm{e}_{\Gamma_{(\mathsf{k})}}$
    $\Gamma = \Gamma_{(\mathsf{m})}$
    $\mu^{(0)} = \mu_{(\mathsf{m})}^{(\mathsf{i}_{\max})}$
  end for
  $\mathsf{i}_{\max} = 10^6$
  $\hat{\mathbf{X}} = \mathcal{A}(\mathbf{Y}, \Gamma, \alpha, \mu^{(0)}, \varepsilon, \beta, \mathsf{i}_{\max})$
  return $\hat{\mathbf{X}}$

E. Outer loop

Let $\left\{\Gamma_{(1)}, \Gamma_{(2)}, \ldots, \Gamma_{(\mathsf{k})}, \ldots \Gamma_{(\mathsf{g})}\right\}$ be a predefined set of TF patterns. The "outer loop" consists in evaluating $\hat{\mathbf{X}}_{(\mathsf{k})} = \mathcal{A}(\mathbf{Y}, \Gamma_{(\mathsf{k})}, \alpha, \mu_{(\mathsf{k})}^{(0)}, \varepsilon, \beta, \mathsf{i}_{\max})$ for different $\Gamma_{(\mathsf{k})}$, with $\mathsf{i}_{\max}$ set to a small value (*e.g.* $= 10$) and β the user-defined relative accuracy.

The choice of α and $\mu_{(\mathsf{k})}^{(0)}$ seems important as they tell the inner loop algorithm how "aggressively" to perform regularization. We set $\mu_{(\mathsf{k})}^{(0)} = \|\Gamma_{(\mathsf{k})}\|_0 \times \|\mathrm{vec}(\mathbf{Y})\|_\infty$, where $\mathrm{vec}(\cdot)$ vectorizes the matrix. We also set $\alpha = \min\left(\frac{\sigma}{\sqrt{\mathrm{var}(\mathrm{vec}(\mathbf{Y}))}}, 0.99\right)$. The choice of α reflects the "instantaneous" SNR in the region being processed. The choice of these parameters seems to be compatible with the numerous audio examples described in III and could probably be validated on others.

Having computed all $\hat{\mathbf{X}}_{(\mathsf{k})}$ for $\mathsf{k} \in [1..\mathsf{g}]$ (thus for all possible $\Gamma_{(\mathsf{k})}$), we evaluate the (empirical) entropy of each residual $\mathbf{R}_{(\mathsf{k})} := \mathbf{A}\hat{\mathbf{X}}_{(\mathsf{k})} - \mathbf{A}\mathbf{Y}$:

$$\mathrm{e}_{\Gamma_{(\mathsf{k})}} = -\sum_{\mathsf{q}=1}^{\mathsf{Q}} \hat{\mathrm{p}}_{\mathsf{q}} \log_2(\hat{\mathrm{p}}_{\mathsf{q}}), \qquad (3)$$

Table I
AUDASCITY PARAMETERS

Parameters	Segment size [samples]	Overlap [%]	Window	Overlapping segments	Outer loop maximum iterations	Accuracy	Analysis operator	Set of TF patterns	Bins for $e_{\Gamma_{(k)}}$ calculation
Value	$L = 1024$	75	Hamming	$b = 5$	$i_{max} = 10$	$\beta = 10^{-3}$	$\mathbf{A}$ = DFT	See Fig. 1	$Q = \frac{2b+1 \times L}{20}$

where $\hat{p}$ is the empirical probability distribution of the magnitude of the entries of $\mathbf{R}_{(k)}$, estimated by Q bins of normalized histograms. The idea is that the best estimate $\hat{\mathbf{X}}_{(k)}$ produces the residual of least informative content. In the case of Additive White Gaussian Noise (AWGN), which is by construction flat across the spectrogram, this choice makes an obvious sense. Thus, we select the pattern $\Gamma_{(m)}$ which yields the highest entropy $e_{\Gamma_{(k)}}$ for the considered signal region. Since we set low i_{max}, the evaluation procedure is quite fast, and adds only $(g-1)i_{max}$ iterations compared to the case where only one pattern is considered. For the chosen Γ, we can continue iterating from $i = i_{max} + 1$ (using $(\hat{\mathbf{X}}_{(k)}, \mu_{(k)}^{(i_{max})})$ as the initial point) until reaching β-defined convergence.

The overall functioning of the AUDASCITY algorithm is described by the pseudo code in Algorithm 2 for a given block of adjacent frames $\mathbf{Z} \in \mathbb{R}^{L \times (2b+1)}$.

F. Post-processing and overlap-add synthesis

Applying Algorithm 2 to the n^{th} noisy matrix $\mathbf{Y}_n$ yields an estimate $\hat{\mathbf{X}}_n$. We extract its middle column $\hat{\mathbf{x}}_n$ (the central column of the block, corresponding to the segment $\mathbf{y}_n$) (see Fig. 2). This estimate $\hat{\mathbf{x}}_n$ is stored in memory, and later used to perform overlap-add synthesis, which yields the denoised signal. Similarly to the Block-Thresholding algorithm [4] which we use as a comparison in the experimental section, we embed a post-processing step. This last action performs a simple frequency-domain Wiener filtering on $\hat{\mathbf{x}}_n$ before overlap-add synthesis. It uses σ^2 as the estimated noise power and the squared magnitudes of $\mathbf{A}\hat{\mathbf{x}}_n$ as signal power. Practically, we see that this post-processing is useful at very low SNR (*i.e* 0 dB) where we report "musical noise" effect.

G. Backward-caching scheme

To speed up computations, and possibly to account for previously processed information, we incorporate a "backward-caching scheme". This simply means that instead of processing data blocks indexed by $[n-b, n+b]$, we consider only the ones linked to indexes in $[n, n+b]$, assuming that the frames in the interval $[n-b, n-1]$ have been already recovered. These blocks, denoted by $\hat{\mathbf{X}}_{bw}$, are then given to the algorithm, but considered as fixed: the PEW shrinkage in the $\mathbf{Z}$-update step is applied to the augmented matrix $[\mathbf{A}\hat{\mathbf{x}}_{bw}\ \mathbf{A}\mathbf{X}^{(i-1)} + \mathbf{U}^{(i-1)}]$. Also, when computing the empirical entropy, we account for the $\hat{\mathbf{X}}_{bw}$ blocks.

We have seen that, in practice, this has a negligible effect on the recovery performance, while indeed reducing the computational cost (see III-B for details).

III. EXPERIMENTS

We conduct experiments on the MPEG items and the RWC Music Database [12]. On the latter, we use the "Pop" and "Jazz" genres as is, and subcategorize the "Classic" genre (Vocals, Chamber, Symphonies), leading to 6 subsets. All the tracks are sufficiently diverse to reflect the robustness of the approach on different audio content. We contaminate the audio tracks with additive white gaussian noise at five SNR levels. The AUDASCITY algorithm is confronted to the state-of-the-art time-frequency block thresholding (BT). We note that BT is parameterized with true variance σ^2. AUDASCITY parameters are as listed in table I. In this study, we choose a non redundant DFT transform for $\mathbf{A}$ *i.e.* $P = L$. We conduct three phases in testings: one to analyze SNR numerical results, a second to look at performance on sound classification. Finally, we compared perceptual differences in a listening test.

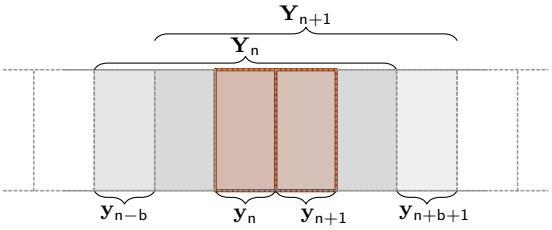
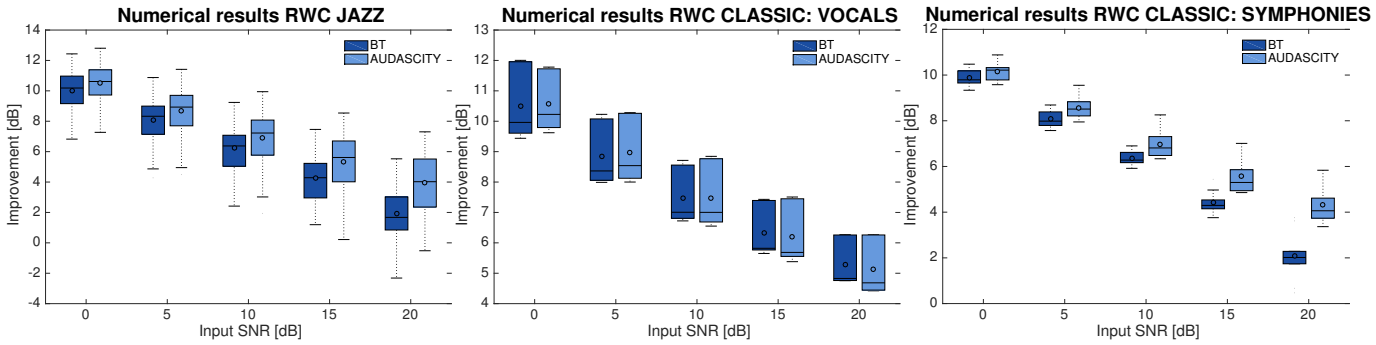


Figure 2. Segment processing for frame n and frame $n+1$

A. Numerical Results

We consider five input SNR levels in dB: $\{0, 5, 10, 15, 20\}$. The different stencils $\Gamma_{(k)}$ are listed in Fig. 1 with rows corresponding to the frequency dimension and columns accounting for the time dimension. $\Gamma_{(1)}$ emphasizes tonal content while $\Gamma_{(2)}$ is more suitable for transients and attacks. $\Gamma_{(3)}$ and $\Gamma_{(4)}$ are more likely to stress tonal transitions. $\Gamma_{(5)}$ is designed to avoid pre-echo artifacts as in [13]. Finally, $\Gamma_{(6)}$ is a default stencil foreseen to replace the others when no particular structure is identified.

Fig. 3 shows the average improvement as a function of input SNR (averaged over all audio tracks). We confirm that for every setting with SNR above 0 dB and for all audio tracks in RWC Jazz, Classic Chamber and Symphonies, our approach is better in average in terms of numerical improvement (statistical descriptors: p-values from t-tests are listed in table II where smaller value means higher significance. Bold results underline significant difference in performance with at least a 5% confidence interval. As expected, the difference in performance increases with the input SNR: this occurs because BT relies strongly on the noise model, while we are trying to "emphasize" the signal instead (more precisely, its TF behavior). One hypothesis is that this model would be robust enough to account for different types of stationary noise (not only AWGN), but it would most probably require different parameterization (*e.g.*, changing the way the data fidelity parameter ε is computed). We remark that another work [13] based on social sparsity reported underperforming in SNR compared to BT. We can also note that some results



(a) RWC Jazz (b) RWC Classic: Opera Singing (c) RWC Classic: Symphonies

Figure 3. Numerical Results: SNR improvement [dB]

in [14] reveal usefulness of social priors on top of cosparse models.

Table II
T-TESTS P-VALUES: AUDASCITY VS. BT STATISTICAL COMPARISON

SNR [dB]	Vocals	Chamber	Symphonies	Jazz	Pop	MPEG
0	0.9074	0.0516	0.1771	0.0581	**3.96 e-5**	0.96
5	0.8564	**0.0239**	**0.0483**	0.0517	**0.0024**	0.94
10	0.9904	0.0096	**0.0166**	**0.0426**	0.1110	0.97
15	0.8507	**0.0011**	**0.0015**	**0.0029**	0.31	0.85
20	0.7503	**2.3 e-8**	**0.0001**	**6.72 e-7**	**2.19 e-7**	0.8292

B. Computational Aspects

To quantify the speed performance of AUDASCITY we perform run time tests on a laptop computer with 2.8 GHz Intel Core i7 processor and 16 GB ram memory. All tests are run on 30 second audio excerpts sampled at 16 kHz with Matlab® in single thread mode. Table III displays all the results. The computational cost is, as the improvement, strongly linked to the input SNR. Indeed, as the SNR decreases, the inner loop takes more iterations to converge. On the other hand, runtime benefits clearly from the backward-caching scheme, and it appears that we can reach a 30% speed-up without performance loss. These encouraging results make it possible to envision real-time applications even for moderate SNR.

Table III
AUDASCITY RUNTIME PERFORMANCES

	Input SNR [dB]	0	5	10	15	20
Backward-caching on	Runtime [s]	170.6	118.0	79.0	57.3	46.6
	Improvement [dB]	9.24	6.35	3.64	1.41	0.14
Backward-caching off	Runtime [s]	273.3	194.9	135.9	98.5	75.1
	Improvement [dB]	9.28	6.58	4.02	1.98	0.66

C. Classification Results

One example for usefulness of a performant denoising is classification of audio signals, where the noise pollution, generally, has adversarial effect on performance. The extent to which the performance is affected depends on several factors, among them the type of extracted features being an important one. For this reason, we conduct a small scale classification benchmark, involving two types of features: i) widely used Mel Frequency Cepstral Coefficients (MFCC), and ii) the features based on the (second order) *scattering transform*, recently proposed by Mallat et al. [15]. The latter has been shown to outperform MFCC on several occasions, owing to its ability to exploit information from a wider frequency range [16].

We use the training database provided by DCASE 2016 sound event recognition challenge [17]. The database split in 15 classes, is randomly partitioned (balanced classes), such that 70% of audio files are used for training, and the remaining 30% are used as test data. The chosen database enables us to directly evaluate the effect of SNR and denoising on classification performance. The following experiments are performed using a kernel-based SVM classifier [18], and the features are extracted from overlapping audio segments of duration $\mathrm{T} \approx 0.18$s. For scattering, we use *Scatnet* toolbox provided by the authors [19].

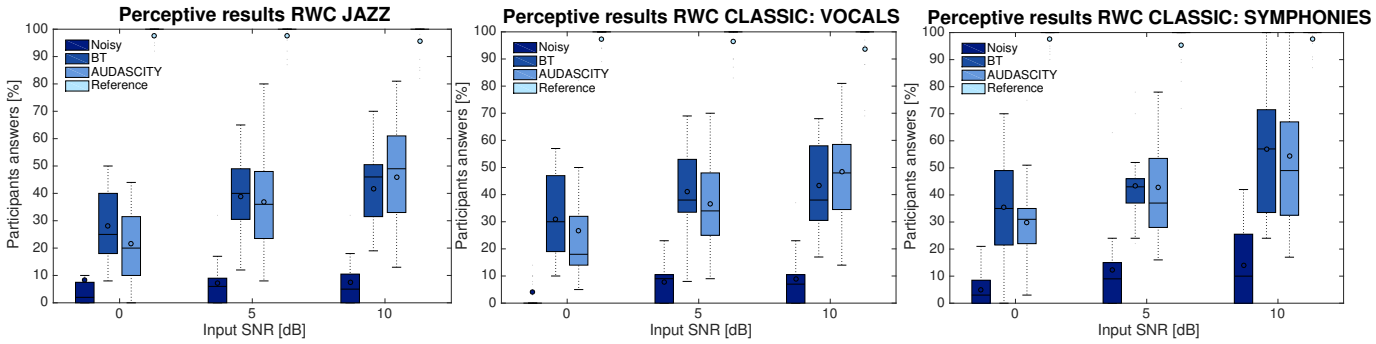
Table IV presents the classification accuracy for the noiseless, noisy (SNR $= 0$dB) and denoised audio, with the training and test sets equally treated. The negative impact of low SNR on accuracy can be clearly seen, particularly for the MFCC features. Likewise, denoising leads to an improvement in performance, with the proposed algorithm outperforming state-of-the-art BT, for both types of features.

Table IV
CLASSIFICATION ACCURACY

	Noiseless	Audascity	BT	Noisy
MFCC	78%	68%	65%	63%
Scattering	93%	89%	86%	83%

D. Perceptual Results

While the objective assessment is certainly an important quality metric, the usefulness of a performant denoising may also include the auditory appearance. In this way, in this part, we are interested in the perceptual quality of the denoising procedure. For that purpose, we use a listening test available online through the application framework presented in [20]. This procedure is based on the MUltiple Stimuli with Hidden Reference and Anchor (MUSHRA) evaluation framework [21]. We evaluate a subset of the conditions presented for the numerical experiments in Section III-A. For the test not to last longer than 40 minutes, we choose 3 input SNR $\{0,\ 5,\ 10$ dB$\}$. As for the sound material, we pick a 4 seconds excerpt randomly chosen from each genre of the RWC database and the all MPEG database leading to a total



(a) RWC Jazz (b) RWC Classic: Opera singing (c) RWC Classic: Symphonies

Figure 4. Perceptual Results: MUSHRA Test

number of 39 conditions to test. 20 participants take part in the listening experiment and are asked to rate the similarity between the clean reference and the processed signals on a scale from 0 to 100. Figure 4 displays results for the listening experiment. The high variability between participants does not bring a clear superiority of one or another algorithm. For quite adversarial conditions (0 and 5dB) we can see a slight advantage for BT while as the SNR increases, the trend tends to reverse. For lowest SNR settings, participants reported a pronounced "musical noise" effect. Some complementary techniques like multichannel Wiener filter or sequential denoising [22] could be investigated in a further study to address this issue. Besides, during informal pilot listening tests, we noticed that the AUDASCITY estimate feels "richer", in the sense that it recovers both low and high frequency content (as opposed to BT which is severely low-pass filtered). Even though the perceptual quality is not always following the trend of numerical improvements, this study gives promising results. Further improvements and a validation for higher SNRs could be considered in a future study.

IV. Conclusion

We presented a new algorithm to address the problem of audio denoising. This method using "Social Cosparsity", is more efficient on SNR improvement than state-of-the-art. Coupling sparsity structure in time-frequency and analysis sparse priors on the signal to be denoised lets AUDASCITY be fully adaptive and retain low complexity. Subjective evaluation shows encouraging results even if some weaknesses are identified at low signal-to-noise ratios. On the other hand, this new method showed improved classification results when used as a denoising pre-processing block. Future work could envision better perceptual improvements, include colored noise and extend social cosparsity to other audio recovery tasks. A study currently in progress will also feature comparisons with denoising methods using simple cosparse priors.

Acknowledgment

The authors thank Matthieu Kowalski for insightful discussions and advice. This work was supported in part by the European Research Council, PLEASE project (ERC-StG-2011-277906).

References

[1] R. McAulay and M. Malpass, "Speech enhancement using a soft-decision noise suppression filter," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 2, pp. 137–145, Apr 1980.

[2] Y. Ephraim and D. Malah, "Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 6, pp. 1109–1121, Dec 1984.

[3] S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 27, no. 2, pp. 113–120, Apr 1979.

[4] G. Yu, S. Mallat, and E. Bacry, "Audio denoising by time-frequency block thresholding," *IEEE Transactions on Signal Processing*, vol. 56, no. 5, pp. 1830–1839, 2008.

[5] S. Mallat, *A Wavelet Tour of Signal Processing*. Academic Press, 1999.

[6] S. Kitić, "Cosparse regularization of physics-driven inverse problems," PhD Thesis, IRISA, Inria Rennes, 2015. [Online]. Available: https://hal.archives-ouvertes.fr/tel-01237323

[7] S. Kitić, N. Bertin, and R. Gribonval, "Sparsity and cosparsity for audio declipping: a flexible non-convex approach," in *Latent Variable Analysis and Signal Separation (LVA/ICA)*. Liberec, Czech Republic: Springer, 2015, pp. 243–250.

[8] M. Kowalski, K. Siedenburg, and M. Dorfler, "Social sparsity! neighborhood systems enrich structured shrinkage operators," *IEEE Transactions on Signal Processing*, vol. 61, no. 10, pp. 2498–2511, 2013.

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

[10] K. Siedenburg, M. Kowalski, and M. Dorfler, "Audio declipping with social sparsity," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing*, no. 2, pp. 1577–1581, 2014.

[11] J. Eckstein and D. Bertsekas, "On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, no. 1-3, pp. 293–318, 1992.

[12] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Popular, classical and jazz music databases." in *ISMIR*, vol. 2, 2002, pp. 287–288.

[13] K. Siedenburg and M. Dörfler, "Audio denoising by generalized time-frequency thresholding," in *Audio Engineering Society Conference: 45th International Conference: Applications of Time-Frequency Processing in Audio*. Audio Engineering Society, 2012.

[14] C. Gaultier, S. Kitić, N. Bertin, and R. Gribonval, "Cosparse denoising: The importance of being social," in *The Signal Processing with Adaptive Sparse Structured Representations (SPARS) workshop*, 2017.

[15] J. Bruna and S. Mallat, "Classification with scattering operators," *arXiv preprint arXiv:1011.3023*, 2010.

[16] J. Andén, V. Lostanlen, and S. Mallat, "Joint time-frequency scattering for audio classification," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2015, pp. 1–6.

[17] "DCASE - Detection and Classification of Acoustic Scenes and Events, 2016," http://www.cs.tut.fi/sgn/arg/dcase2016/, accessed: Sept, 2016.

[18] C. Chang and C. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[19] J. Andén, L. Sifre, S. Mallat, M. Kapoko, V. Lostanlen, and E. Oyallon, "Scatnet," Computer Software. Available: http://www. di. ens. fr/data/software/scatnet/., 2014.

[20] S. Kraft and U. Zölzer, "Beaqlejs: Html5 and javascript based framework for the subjective evaluation of audio quality," in *Linux Audio Conference, Karlsruhe, DE*, 2014.

[21] "ITU-R BS.1534-1: Method for the subjective assessment of intermediate quality levels of coding systems."

[22] Y. Dar, A. Bruckstein, M. Elad, and R. Giryes, "Postprocessing of compressed images via sequential denoising," *arXiv preprint arXiv:1510.09041*, 2015.