A PROPER VERSION OF SYNTHESIS-BASED SPARSE AUDIO DECLIPPER

Pavel Záviška* *Pavel Rajmic** *Ondřej Mokrý** *Zdeněk Průša*†

*Signal Processing Laboratory, Brno University of Technology, Brno, Czech Republic
†Acoustics Research Institute, Austrian Academy of Science, Vienna, Austria
Email: *{xzavis01, rajmic, 170583}@vutbr.cz, †zdenek.prusa@oeaw.ac.at

ABSTRACT

Methods based on sparse representation have found great use in the recovery of audio signals degraded by clipping. The state of the art in declipping within the sparsity-based approaches has been achieved by the SPADE algorithm by Kitić et. al. (LVA/ICA'15). Our recent study (LVA/ICA'18) has shown that although the original S-SPADE can be improved such that it converges faster than the A-SPADE, the restoration quality is significantly worse. In the present paper, we propose a new version of S-SPADE. Experiments show that the novel version of S-SPADE outperforms its old version in terms of restoration quality, and that it is comparable with the A-SPADE while being even slightly faster than A-SPADE.

***Index Terms*—** Declipping, Sparse, Cosparse, Synthesis, Analysis

1. INTRODUCTION

Clipping is one of the common types of signal degradation. It is usually caused by an element in the signal path whose dynamic range is insufficient compared to the dynamics of the signal. This fact causes the peaks of the signal to be cut (saturated). More exactly, in the so-called *hard clipping*, samples of the input signal $\mathbf{x} \in \mathbb{R}^N$ that exceed the dynamic range given by the thresholds $[-\theta_c, \theta_c]$ are modified such that the signal output can be described by the formula

$$\mathbf{y}[n] = \begin{cases} \mathbf{x}[n] & \text{for} \quad |\mathbf{x}[n]| < \theta_c, \\ \theta_c \cdot \operatorname{sgn}(\mathbf{x}[n]) & \text{for} \quad |\mathbf{x}[n]| \geq \theta_c. \end{cases} \tag{1}$$

Due to the great number of higher harmonics that appear in the clipped signal, the clipping has a negative effect on the perceived audio quality [1]. Therefore it is inevitable to perform restoration, so-called *declipping*, i.e. a recovery of the clipped samples, based on the observed signal $\mathbf{y}$.

In the hard clipping case, which is the context of the paper, the signal samples can be divided into three sets R, H, and L, which correspond to "reliable" samples and samples that have been clipped to "high" and "low" clipping thresholds, respectively. To select only samples from a specific set, the *restriction operators* M_R, M_H, M_L will be used. These operators can be understood as linear projectors or identity matrices with specific columns removed that correspond to the three cases.

With the additional information that the positive clipped samples should lie above the $\theta_c > 0$ and the negative clipped samples below $-\theta_c$, a set of feasible solutions Γ is defined as a (convex) set of time-domain signals such that

$$\Gamma = \Gamma(\mathbf{y}) = \{\tilde{\mathbf{x}} \mid M_R\tilde{\mathbf{x}} = M_R\mathbf{y}, M_H\tilde{\mathbf{x}} \geq \theta_c, M_L\tilde{\mathbf{x}} \leq -\theta_c\}. \tag{2}$$

The restored signal, $\hat{\mathbf{x}}$, is naturally required to be a member of the set, i.e. $\hat{\mathbf{x}} \in \Gamma$. Finding $\hat{\mathbf{x}}$ is an ill-posed problem since there are infinitely many possible solutions. A possible way to treat this problem is to exploit the fact that audio signals are sparse with respect to a (time-)frequency transform. In other words, the goal is to find the signal $\hat{\mathbf{x}}$ from the set Γ of the highest sparsity.

In the past, several approaches to declipping were introduced. Focusing on the sparsity-based methods, the very first method using the sparsity assumption was reported in [2]; it was based on the greedy approximation of a signal within the reliable parts. In [3], convex optimization was used. According to [4], adding the structure to the coefficients may lead to the improvement in the restoration quality. The authors of [5] used an iterative hard thresholding algorithm that was constrained to solve the declipping task and in [6] reformulated the task to the analysis approach to the sparsity.

On top of these approaches, non-negative matrix factorization has also been recently adopted to audio declipping [7].

As far as the authors know, [8] presented the current state-of-the-art, a heuristic declipping algorithm for both the analysis and the synthesis models (the SPADE algorithm). Until recently, the synthesis variant was considered significantly slower due to the difficult projection step, but [9] has shown that the opposite is true—the acceleration makes the synthesis model require even fewer iterations to converge, making it faster than the analysis variant. Unfortunately, the restoration quality of the synthesis variant has been shown to be substantially worse. In this paper, the problem of the original

The authors thank S. Kitić for providing them with his implementation of the SPADE algorithms and for discussion. The work was supported by the joint project of the FWF and the Czech Science Foundation (GAČR): numbers I 3067-N30 and 17-33798L, respectively. Research described in this paper was financed by the National Sustainability Program under grant LO1401. Infrastructure of the SIX Center was used.

synthesis variant is briefly explained and a new, more proper, synthesis version of the algorithm is presented.

2. SPADE ALGORITHMS

SPADE (SParse Audio DEclipper) [8] by Kitić et. al. is a sparsity-based heuristic declipping algorithm. It is derived using the Alternating Direction Method of Multipliers (ADMM), which is briefly revised first. For details and proofs, see [10].

2.1. ADMM

The ADMM [11] is a means for solving problems of the form $\min f(\mathbf{x}) + g(A\mathbf{x})$, or equivalently

$$\min_{\mathbf{x},\mathbf{z}} f(\mathbf{x}) + g(\mathbf{z}) \quad \text{s.t.} \quad A\mathbf{x} - \mathbf{z} = 0, \tag{3}$$

where $\mathbf{x} \in \mathbb{C}^N, \mathbf{z} \in \mathbb{C}^P$ and $A : \mathbb{C}^N \rightarrow \mathbb{C}^P$ is a linear operator. ADMM is based on minimizing the Augmented Lagrangian, defined for (3) as:

$$L_\rho(\mathbf{x},\mathbf{y},\mathbf{z}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^\top(A\mathbf{x}-\mathbf{z}) + \frac{\rho}{2}\|A\mathbf{x}-\mathbf{z}\|_2^2, \tag{4}$$

where $\rho > 0$ is called the *penalty parameter*. The ADMM consists of three steps: minimization of (4) over $\mathbf{x}$, over $\mathbf{z}$, and the update of the dual variable, formally [11]:

$$\mathbf{x}^{(i+1)} = \arg\min_{\mathbf{x}} L_\rho\left(\mathbf{x}, \mathbf{z}^{(i)}, \mathbf{y}^{(i)}\right), \tag{5a}$$

$$\mathbf{z}^{(i+1)} = \arg\min_{\mathbf{z}} L_\rho\left(\mathbf{x}^{(i+1)}, \mathbf{z}, \mathbf{y}^{(i)}\right), \tag{5b}$$

$$\mathbf{y}^{(i+1)} = \mathbf{y}^{(i)} + \rho\left(A\mathbf{x}^{(i+1)} + \mathbf{z}^{(i+1)}\right). \tag{5c}$$

The ADMM can be often seen in the so-called *scaled form*, which we obtain by substituting a dual variable $\mathbf{y}$ with the scaled dual variable $\mathbf{u} = \mathbf{y}/\rho$.

2.2. SPADE

The SPADE algorithm [8] approximates the solution of the following NP-hard regularized inverse problems

$$\min_{\mathbf{x},\mathbf{z}} \|\mathbf{z}\|_0 \text{ s.t. } \mathbf{x} \in \Gamma(\mathbf{y}) \text{ and } \|A\mathbf{x} - \mathbf{z}\|_2 \leq \epsilon, \tag{6a}$$

$$\min_{\mathbf{x},\mathbf{z}} \|\mathbf{z}\|_0 \text{ s.t. } \mathbf{x} \in \Gamma(\mathbf{y}) \text{ and } \|\mathbf{x} - D\mathbf{z}\|_2 \leq \epsilon, \tag{6b}$$

where (6a) and (6b) represent the problem formulation for the analysis and the synthesis variant, respectively. Here, Γ denotes the set of feasible solutions (see Eq. (2)), $\mathbf{x} \in \mathbb{R}^N$ stands for the unknown signal in the time domain, and $\mathbf{z} \in \mathbb{C}^P$ contains the (also unknown) coefficients. As for the linear operators, $A : \mathbb{R}^N \rightarrow \mathbb{C}^P$ is the analysis (thus $P \geq N$) and $D : \mathbb{C}^P \rightarrow \mathbb{R}^N$ is the synthesis, while it holds $D = A^*$. For computational reasons, we restrict ourselves only to the Parseval tight frames [12], i.e. $DD^* = A^*A = Id$, with unitary operators as their special cases.

The problems (6) can be recast as the sum of two indicator functions:

$$\arg\min_{\mathbf{x},\mathbf{z},k} \iota_\Gamma(\mathbf{x}) + \iota_{\ell_0 \leq k}(\mathbf{z}) \text{ s. t. } \begin{cases} \|A\mathbf{x} - \mathbf{z}\|_2 \leq \epsilon, \\ \|\mathbf{x} - D\mathbf{z}\|_2 \leq \epsilon, \end{cases} \tag{7}$$

where $\iota_\Gamma(\mathbf{x})$ makes the restored signal lie in Γ and $\iota_{\ell_0 \leq k}(\mathbf{z})$ is a shorthand notation for $\iota_{\{\tilde{\mathbf{z}} \,|\, \|\tilde{\mathbf{z}}\|_0 \leq k\}}(\mathbf{z})$, which enforces the sparsity of the coefficients.

The signal is cut into overlapping blocks and windowed prior to processing. Therefore, in (7), $\mathbf{y}$ should be understood as one of the signal chunks. The overall resulting signal is made up by the overlap-add procedure. As the transformations, [8] uses an overcomplete DFT and IDFT, respectively.

2.3. A-SPADE

To solve the analysis variant of (7), the Augmented Lagrangian is formed according to (4) and the three ADMM steps according to (5) are constructed:

$$\mathbf{x}^{(i+1)} = \arg\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{z}^{(i)} + \mathbf{u}^{(i)}\|_2^2 \quad \text{s.t. } \mathbf{x} \in \Gamma, \tag{8a}$$

$$\mathbf{z}^{(i+1)} = \arg\min_{\mathbf{z}} \|A\mathbf{x}^{(i+1)} - \mathbf{z} + \mathbf{u}^{(i)}\|_2^2 \text{ s.t. } \|\mathbf{z}\|_0 \leq k, \tag{8b}$$

$$\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + A\mathbf{x}^{(i+1)} - \mathbf{z}^{(i+1)}. \tag{8c}$$

The report [10] shows in detail that the subproblem (8a) is, in fact, a projection of $(A^*(\mathbf{z}^{(i)} + \mathbf{u}^{(i)}))$ onto Γ, efficiently implemented as an elementwise mapping in the time domain [8, 9]. Furthermore, the solution of (8b) is obtained by applying the hard-thresholding operator $\mathcal{H}_k$ to $(A\mathbf{x}^{(i+1)} + \mathbf{u}^{(i)})$, setting all but k its largest elements to zero, taking into account the complex conjugate coefficients. The A-SPADE algorithm is finally obtained by adding the sparsity relaxation step to the above steps (8), in which the sparsity of the representation is allowed to increase during iterations. See Alg. 1.

2.4. S-SPADE original and S-SPADE new

In the synthesis variant, the situation is different. Alg. 2 presents the S-SPADE algorithm from [8]. Here, the two minimization steps are both carried over $\mathbf{z}$. Although this approach is based on the ADMM, it is explained in [10] that this algorithm solves a problem that is different from (6b). Therefore the original S-SPADE is not really a synthesis counterpart of the A-SPADE. The report [10] shows that only with unitary operators ($A = D^{-1}$) do all the three problems coincide.

Next, we show how the synthesis variant of the SPADE algorithm is derived such that it indeed solves (6b). First of all, the problem (7) is altered as

$$\arg\min_{\mathbf{x},\mathbf{z},k} \iota_\Gamma(\mathbf{x}) + \iota_{\ell_0 \leq k}(\mathbf{z}) \text{ s. t. } D\mathbf{z} - \mathbf{x} = 0. \tag{9}$$

Algorithm 1: A-SPADE from [8]

```
Require: $A, \mathbf{y}, M_{\mathrm{R}}, M_{\mathrm{H}}, M_{\mathrm{L}}, s, r, \epsilon$
1  $\hat{\mathbf{x}}^{(0)} = \mathbf{y}, \mathbf{u}^{(0)} = \mathbf{0}, i = 0, k = s$
2  $\bar{\mathbf{z}}^{(i+1)} = \mathcal{H}_k\left(A\hat{\mathbf{x}}^{(i)} + \mathbf{u}^{(i)}\right)$
3  $\hat{\mathbf{x}}^{(i+1)} = \arg\min_{\mathbf{x}} \|A\mathbf{x} - \bar{\mathbf{z}}^{(i+1)} + \mathbf{u}^{(i)}\|_2^2$
     s.t. $\mathbf{x} \in \Gamma$
4  if $\|A\hat{\mathbf{x}}^{(i+1)} - \bar{\mathbf{z}}^{(i+1)}\|_2 \le \epsilon$ then
5      terminate
6  else
7      $\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + A\hat{\mathbf{x}}^{(i+1)} - \bar{\mathbf{z}}^{(i+1)}$
8      $i \leftarrow i + 1$
9      if $i \bmod r = 0$ then
10         $k \leftarrow k + s$
11     end
12     go to 2
13 end
14 return $\hat{\mathbf{x}} = \hat{\mathbf{x}}^{(i+1)}$
```

Algorithm 2: S-SPADE from [8]

```
Require: $D, \mathbf{y}, M_{\mathrm{R}}, M_{\mathrm{H}}, M_{\mathrm{L}}, s, r, \epsilon$
1  $\hat{\mathbf{z}}^{(0)} = D^*\mathbf{y}, \mathbf{u}^{(0)} = \mathbf{0}, i = 0, k = s$
2  $\bar{\mathbf{z}}^{(i+1)} = \mathcal{H}_k\left(\hat{\mathbf{z}}^{(i)} + \mathbf{u}^{(i)}\right)$
3  $\hat{\mathbf{z}}^{(i+1)} = \arg\min_{\mathbf{z}} \|\mathbf{z} - \bar{\mathbf{z}}^{(i+1)} + \mathbf{u}^{(i)}\|_2^2$
     s.t. $D\mathbf{z} \in \Gamma$
4  if $\|\hat{\mathbf{z}}^{(i+1)} - \bar{\mathbf{z}}^{(i+1)}\|_2 \le \epsilon$ then
5      terminate
6  else
7      $\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + \hat{\mathbf{z}}^{(i+1)} - \bar{\mathbf{z}}^{(i+1)}$
8      $i \leftarrow i + 1$
9      if $i \bmod r = 0$ then
10         $k \leftarrow k + s$
11     end
12     go to 2
13 end
14 return $\hat{\mathbf{x}} = D\hat{\mathbf{z}}^{(i+1)}$
```

Algorithm 3: S-SPADE proposed

```
Require: $D, \mathbf{y}, M_{\mathrm{R}}, M_{\mathrm{H}}, M_{\mathrm{L}}, s, r, \epsilon$
1  $\hat{\mathbf{x}}^{(0)} = D^*\mathbf{y}, \mathbf{u}^{(0)} = \mathbf{0}, i = 1, k = s$
2  $\bar{\mathbf{z}}^{(i+1)} = \mathcal{H}_k\left(D^*(\hat{\mathbf{x}}^{(i)} - \mathbf{u}^{(i)})\right)$
3  $\hat{\mathbf{x}}^{(i+1)} = \arg\min_{\mathbf{x}} \|D\bar{\mathbf{z}}^{(i+1)} - \mathbf{x} + \mathbf{u}^{(i)}\|_2^2$
     s.t. $\mathbf{x} \in \Gamma$
4  if $\|D\bar{\mathbf{z}}^{(i+1)} - \hat{\mathbf{x}}^{(i+1)}\|_2 \le \epsilon$ then
5      terminate
6  else
7      $\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + D\bar{\mathbf{z}}^{(i+1)} - \hat{\mathbf{x}}^{(i+1)}$
8      $i \leftarrow i + 1$
9      if $i \bmod r = 0$ then
10         $k \leftarrow k + s$
11     end
12     go to 2
13 end
14 return $\hat{\mathbf{x}} = \hat{\mathbf{x}}^{(i+1)}$
```

Next, the Augmented Lagrangian is formed,

$$L_\rho(\mathbf{x},\mathbf{y},\mathbf{z}) = \iota_{\ell_0 \le k}(\mathbf{z}) + \iota_\Gamma(\mathbf{x}) + \mathbf{y}^\top(D\mathbf{z}-\mathbf{x}) + \frac{\rho}{2}\|D\mathbf{z}-\mathbf{x}\|_2^2. \tag{10}$$

Using the scaled form, (10) appears as

$$L_\rho(\mathbf{x},\mathbf{z},\mathbf{u}) = \iota_{\ell_0 \le k}(\mathbf{z}) + \iota_\Gamma(\mathbf{x}) + \frac{\rho}{2}\|D\mathbf{z}-\mathbf{x}+\mathbf{u}\|_2^2 - \frac{\rho}{2}\|\mathbf{u}\|_2^2, \tag{11}$$

leading to the following ADMM steps:

$$\mathbf{z}^{(i+1)} = \arg\min_{\mathbf{z}} \|D\mathbf{z} - \mathbf{x}^{(i)} + \mathbf{u}^{(i)}\|_2^2 \text{ s.t. } \|\mathbf{z}\|_0 \le k, \tag{12a}$$

$$\mathbf{x}^{(i+1)} = \arg\min_{\mathbf{x}} \|D\mathbf{z}^{(i+1)} - \mathbf{x} + \mathbf{u}^{(i)}\|_2^2 \text{ s.t. } \mathbf{x} \in \Gamma, \tag{12b}$$

$$\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + D\mathbf{z}^{(i+1)} - \mathbf{x}^{(i+1)}. \tag{12c}$$

As in Sec. 2.2, adding the sparsity relaxation step and a termination criterion leads to the final shape of the proposed S-SPADE algorithm—see Alg. 3.

Unlike the original variant of S-SPADE in [8], where the projection in the frequency domain was required and a special projection lemma had to be used [9], the projection step (12b) in the proposed S-SPADE algorithm is a simple elementwise mapping as is the case of the analysis variant.

The solution of the minimization step (12a) is obtained in Alg. 3 by applying the hard-thresholding $\mathcal{H}_k$ [10]. Note that due to the non-orthogonality of D, such a vector is only an approximate solution to (12a) (in contradiction to A-SPADE where $\mathcal{H}_k$ solves (8b) exactly, cf. Alg. 1).

The computational cost of the SPADE algorithms is dominated by the signal transformations (i.e. the synthesis and analysis). All the three algorithms require precisely one synthesis and one analysis per iteration, and therefore, in theory, the computational complexity of the algorithms is the same.

3. EXPERIMENTS AND RESULTS

The following experiments were designed to compare the proposed variant of S-SPADE (denoted S-SPADE$_{\mathrm{DP}}$ with the index for "done properly") with the original A-SPADE and with the original S-SPADE from [8] (S-SPADE$_{\mathrm{O}}$) in terms of the quality of restoration and the speed of convergence.

Experiments were performed on five diverse audio files with a 16 kHz sampling rate. In the preprocessing step, the signals were peak-normalized and then artificially clipped using multiple clipping thresholds $\theta_{\mathrm{c}} \in \{0.1, \dots, 0.9\}$.

All audio samples were processed frame-wise, using the 1024-sample-long Hann window with 75 % overlap. The algorithms were implemented in MATLAB 2017a using the LTFAT toolbox [13] for signal synthesis and analysis. As the signal transformation, the oversampled DFT is used. The relaxation parameters of all the algorithms were set to $r = 1, s = 1$ and $\epsilon = 0.1$.

The restoration quality was evaluated using ΔSDR, which expresses the signal-to-distortion improvement in dB, defined as $\Delta\mathrm{SDR} = \mathrm{SDR}(\mathbf{x},\hat{\mathbf{x}}) - \mathrm{SDR}(\mathbf{x},\mathbf{y})$, where $\mathbf{y}$ represents the clipped signal, $\mathbf{x}$ is the original undistorted signal and $\hat{\mathbf{x}}$ denotes the restored signal. The SDR itself is computed as:

$$\mathrm{SDR}(\mathbf{u},\mathbf{v}) = 10\log\frac{\|\mathbf{u}\|_2^2}{\|\mathbf{u}-\mathbf{v}\|_2^2} \text{ [dB]}. \tag{13}$$

The advantage of using ΔSDR is that it does not depend on whether the SDR is computed on the whole signal or on the clipped samples only (assuming that the restored signal matches the clipped signal on reliable samples).

Fig. 1 presents the overall ΔSDR results of all SPADE algorithms depending on the clipping threshold θ_{c}. When no redundancy (orthonormal case) is used, all three algorithms perform equally, which results in the black line. With higher

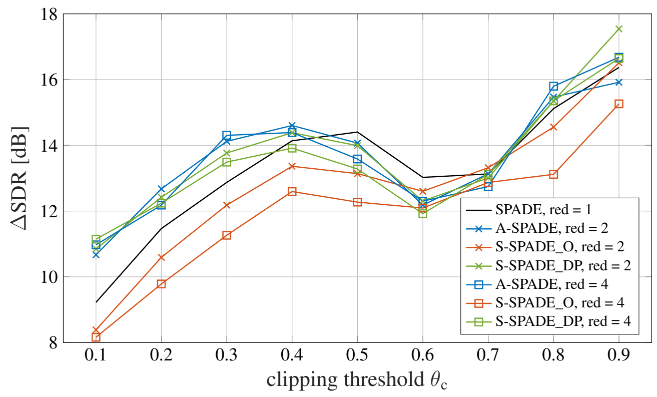

Fig. 1. Average performance in terms of ΔSDR for all three algorithms. Notation "red" denotes redundancy of the DFT.

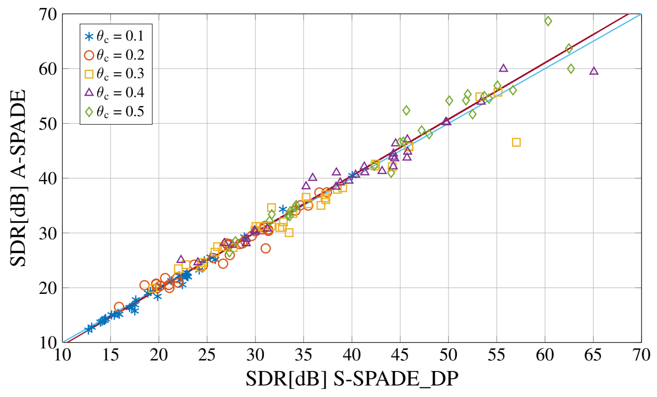

Fig. 3. Scatter plot of SDR values for A-SPADE and S-SPADE$_{DP}$, computed locally on blocks 2048 samples long.

redundancies, both A-SPADE and S-SPADE$_{DP}$ significantly outperform S-SPADE$_O$, especially for lower thresholds θ_c.

Apart from overall ΔSDR evaluation, a more detailed comparison can be seen on scatter plots in Figs. 2 and 3, where S-SPADE$_{DP}$ is compared with S-SPADE$_O$ and A-SPADE, respectively. Each mark in the scatter plot corresponds to the SDR value obtained from a particular 2048-sample-long block. For clarity, only results for clipping thresholds from 0.1 to 0.5 are displayed. Fig. 2 displays the linear regression line; clearly, a majority of the marks are placed below the blue identity line, meaning that in most of the time chunks, the S-SPADE$_{DP}$ performed better than S-SPADE$_O$. Results from the second scatter plot in Fig. 3 prove an on-par restoration quality of A-SPADE and S-SPADE$_{DP}$, where S-SPADE$_{DP}$ performed somewhat better for low SDR and vice versa.

The last experiment compares the SPADE algorithms in terms of the speed of convergence. For this purpose, the number of iterations was fixed for each processed block and the ΔSDR was computed from the whole restored signal. More precisely, the number of iterations varied from 10 to 200 (the termination criterion based on ϵ thus does not come into play). The results are presented in Fig. 4 and they indicate that for redundant operators, S-SPADE$_{DP}$ converges faster than A-SPADE. S-SPADE$_O$ gains the SDR quickly but for a higher number of iterations it is not able to achieve a sufficient ΔSDR.

The source codes and sound signals are available at www.utko.feec.vutbr.cz/ rajmic/software/SPADE-DR.zip.

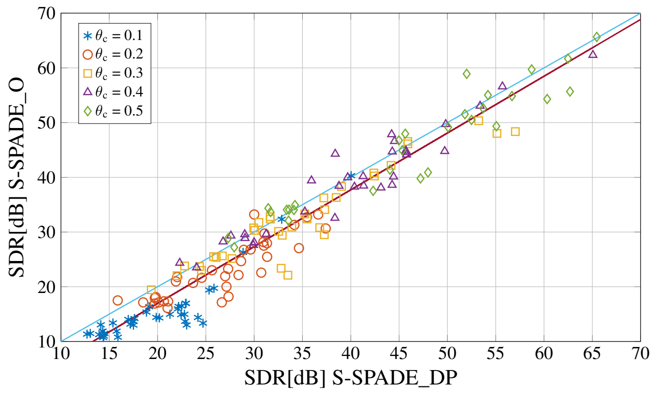

Fig. 2. Scatter plot of SDR values for S-SPADE$_O$ and S-SPADE$_{DP}$, computed locally on blocks 2048 samples long. The blue line is the identity line and the red line represents linear regression. The results shown are for the signal of acoustic guitar with the twice oversampled DFT (red = 2).

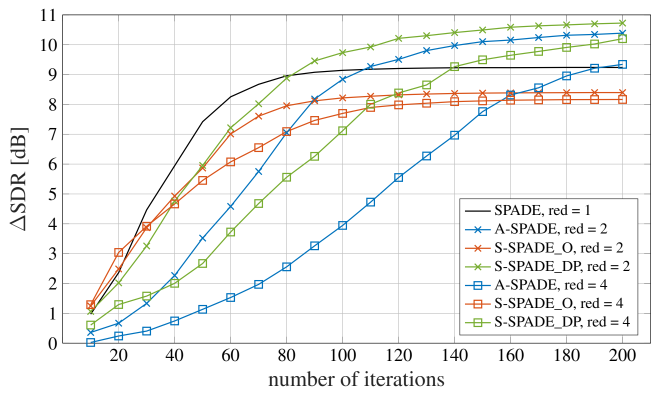

Fig. 4. Average ΔSDR versus the number of iterations.

4. CONCLUSION

A novel algorithm for audio declipping based on the sparse synthesis model was introduced. Unlike the original S-SPADE, the proposed version really solves the problem formulation (6b). The restoration performance is significantly better than with the original version of S-SPADE and it is comparable with the analysis variant. The experiments also show that the new S-SPADE converges faster than A-SPADE.

5. REFERENCES

[1] Ch.-T. Tan, B. C. J. Moore, and N. Zacharov, "The effect of nonlinear distortion on the perceived quality of music and speech signals," *J. Audio Eng. Soc*, vol. 51, no. 11, pp. 1012–1031, 2003.

[2] A. Adler, V. Emiya, M. G. Jafari, M. Elad, R. Gribonval, and M. D. Plumbley, "A constrained matching pursuit approach to audio declipping," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, 2011, pp. 329–332.

[3] A. J. Weinstein and M. B. Wakin, "Recovering a clipped signal in sparseland," *CoRR*, vol. abs/1110.5063, 2011.

[4] K. Siedenburg, M. Kowalski, and M. Dörfler, "Audio declipping with social sparsity," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. 2014, pp. 1577–1581.

[5] S. Kitić, L. Jacques, N. Madhu, M.P. Hopwood, A. Spriet, and C. De Vleeschouwer, "Consistent iterative hard thresholding for signal declipping," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. May 2013, pp. 5939–5943.

[6] S. Kitić, N. Bertin, and R. Gribonval, "Audio declipping by cosparse hard thresholding," in *2nd Traveling Workshop on Interactions between Sparse models and Technology*, 2014.

[7] Ç. Bilen, A. Ozerov, and P. Pérez, "Audio declipping via nonnegative matrix factorization," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2015 IEEE Workshop on*, Oct 2015, pp. 1–5.

[8] S. Kitić, N. Bertin, and R. Gribonval, "Sparsity and cosparsity for audio declipping: a flexible non-convex approach," in *LVA/ICA 2015 – The 12th International Conference on Latent Variable Analysis and Signal Separation*, Liberec, Czech Republic, Aug. 2015.

[9] P. Záviška, P. Rajmic, Z. Průša, and V. Veselý, "Revisiting synthesis model in sparse audio declipper," in *LVA/ICA 2018 – The 14th International Conference on Latent Variable Analysis and Signal Separation*, Guildford, UK, 2018, pp. 429–445, Springer International Publishing.

[10] P. Záviška, O. Mokrý, and P. Rajmic, "S-SPADE Done Right: Detailed Study of the Sparse Audio Declipper Algorithms," techreport, Brno University of Technology, Sept. 2018, URL: `https://arxiv.org/pdf/1809.09847.pdf`.

[11] S. P. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers.," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[12] O. Christensen, *Frames and Bases, An Introductory Course*, Birkhäuser, Boston, 2008.

[13] Z. Průša, P. Søndergaard, P. Balazs, and N. Holighaus, "LTFAT: A Matlab/Octave toolbox for sound processing," in *Proceedings of the 10th International Symposium on Computer Music Multidisciplinary Research (CMMR 2013)*, Marseille, France, October 2013, Laboratoire de Mécanique et d'Acoustique, pp. 299–314, Publications of L.M.A.