Audio Declipping with Unfolded Douglas–Rachford Algorithm

Michal Švento
Dept. of Telecommunications, FEEC
Brno University of Technology
Brno, Czech Republic
michal.svento@vut.cz

***Abstract*—This paper addresses the problem of audio declipping, which occurs when audio signals exceed a certain level, causing distortion and loss of information. To enhance existing methods, we propose a novel solution combining deep unfolding with the Douglas–Rachford algorithm (DRA) within an optimization framework, offering a blend of deep learning and optimization. The declipping problem is formulated as an optimization task that aims to recover the original signal by minimizing sparsity in the time-frequency domain. Our approach transforms each iteration of DRA into a layer of a neural network, optimizing parameters based on training data. Experimental results demonstrate that the unrolled DRA (uDRA) achieves short inference time compared to classical declipping methods, although it does not yet match them in terms of restoration quality. This work highlights the potential of deep unfolding for efficient audio declipping, with future improvements needed to capture the complexities of audio distortion more effectively.**

***Index Terms*—audio inverse problems, deep unfolding, unrolling, declipping**

I. Introduction

Saturation is a common type of nonlinear audio degradation where the signal amplitude above a certain threshold is truncated. This results in harsh artifacts and a reduction of the dynamic range. Even when the distortion is intentional, such as in guitar effects, we usually seek to recover the lost signal information. The process of restoring a signal from its clipped observation is known as declipping.

The first approaches to solving such a task were model-based, assuming and forcing certain physical properties of the signal in the related inverse problem. The sparsity of audio signals in the time-frequency domain of the discrete Gabor transform (DGT) has achieved the state-of-the-art performance [1]–[6]. Despite their good performance, the major drawback of model-based algorithms is their very slow inference and still not plausible results for high distortion [7].

The era of data-based processing introduced many new, successful algorithms for solving audio inverse problems. Their strength is largely due to the availability of extensive training data and shorter inference times compared to traditional digital signal processing (DSP) techniques. However, neural networks lack higher interpretability, operate as black box mappers from distorted to clean signals, and often fail on unseen data.

Generative probabilistic models solve the inverse problem from the Bayesian perspective and have very good perceptual results [8]–[10], since they want to restore the signal to be in high density regions based on distorted observation. Despite the good perceptual results, the high-end GPU is needed for the comparable inference time with traditional DSP methods [9], which makes them also relatively slow.

The deep unfolding (DU), also known as algorithm unrolling, bridges model-based and data-driven approaches by transforming classical iterative optimization algorithms into structured neural networks with learnable parameters [7], [11]. This framework not only enhances computational efficiency but also enables data-driven adaptation while maintaining interpretability. A conceptually similar approach is found in Plug-and-Play (PnP) methods, which incorporate advanced denoising priors within iterative optimization schemes to solve inverse problems [12]–[14]. Both DU and PnP exploit the iterative nature of optimization, demonstrating how classical techniques can be enhanced by modern learning-based components.

Despite relatively large research regarding DU in image processing [15]–[17], to the best knowledge of the author, the DU is underexplored in the audio inverse problems [18]–[20], especially for nonlinear problems. In this paper the Douglas–Rachford algorithm (DRA) will be unrolled for the audio declipping task, where we expect that trainable parameters of DRA will enhance the success of the algorithm.

The rest of the paper is organized as follows. Section II formulates the problem and DU for the minimization problem. In Section III training and evaluation of the unrolled network are described. In the Section IV the performance of the modified algorithm and future improvements for the unfolding framework are discussed.

II. Method

A. Problem Formulation

Let $\mathbf{x}$ be an undistorted audio signal of length N, and consider a nonlinear distortion known as *hard clipping* with a threshold $\theta \in [0, 1]$. The hard clipping operation can be

The work was supported by the Czech Science Foundation (GAČR) Project No. 23-07294S. The author is grateful to NVIDIA for donation of the Titan XP graphic card, which has been used in this research. PhD study of Michal Švento is supervised by Pavel Rajmic. The author thanks for valuable comments and improvements from Pavel Rajmic, Ondřej Mokrý and Eloi Moliner.

expressed as the following element-wise mapping function $f_\theta(\mathbf{x}) \mapsto \mathbf{y}$:

$$y_n = \begin{cases} x_n & \text{if } |x_n| \leq \theta \\ \theta \cdot \operatorname{sgn}(x_n) & \text{if } |x_n| > \theta. \end{cases} \tag{1}$$

To formalize the declipping problem, we define three disjoint index sets: R (reliable samples) corresponding to values that remain unchanged after clipping, H (high-clipped samples) corresponding to values that were clipped to $+\theta$, L (low-clipped samples) corresponding to values that were clipped to $-\theta$.

Using these sets, we may define the convex set $\Gamma \subset \mathbb{C}^P$ of feasible solutions in the time-frequency domain as

$$\Gamma = \{\mathbf{c} \mid (D\mathbf{c})(R) = \mathbf{y}(R), (D\mathbf{c})(H) \geq \theta, (D\mathbf{c})(L) \leq -\theta\}, \tag{2}$$

The operator D is a synthesis operator, specifically the inverse discrete Gabor transform (DGT). As the analysis operator, D^*, the forward DGT is used such that these two operators satisfy the Parseval tight frame condition, $\mathbf{c} = D^*D\mathbf{c}$, where $\mathbf{c}$ are time-frequency coefficients.

The declipping problem is then formulated as the following optimization problem:

$$\arg\min_{\mathbf{c}} \|\mathbf{w} \odot \mathbf{c}\|_1 + \iota_\Gamma(\mathbf{c}), \tag{3}$$

where the first term promotes sparsity in the transform domain using a weighted ℓ_1-norm on the DGT coefficients $\mathbf{c}$, and $\iota_\Gamma(\mathbf{c})$ is the indicator function of the feasible set Γ, enforcing consistency with the observed clipped signal. This formulation is based on the assumption that the ℓ_1-norm of the DGT coefficients of the clean signal $\hat{\mathbf{x}}$ is sparser than that of the clipped signal $\mathbf{y}$, making sparsity minimization a useful constraint to recover the original signal.

B. Algorithmic solution

A well-known approach for solving problem (3) is the DRA [21]. In this work, we consider a version of DRA tailored to audio declipping, as proposed in [2].

Algorithm 1 Unfolded DRA

Require: initializze $\mathbf{c}^{(1)} \in \mathbb{C}^P$, weights $\mathbf{w} \in \mathbb{R}^P$.
L layers (iterations), $\lambda = 1$, $\gamma > 0$

1: **for** $l = 1, 2, \ldots, L$ **do**
2: $\quad \hat{\mathbf{c}}^{(l)} = \operatorname{proj}_\Gamma \mathbf{c}^{(l)}$
3: $\quad \mathbf{c}^{(l+1)} = \mathbf{c}^{(l)} + \lambda(\operatorname{soft}_{\gamma\mathbf{w}}(2\hat{\mathbf{c}}^{(l)} - \mathbf{c}^{(l)}) - \hat{\mathbf{c}}^{(i)})$
return $\hat{\mathbf{x}} = D\hat{\mathbf{c}}^{(l)}$

The **projection step** (line 2) ensures that the iterates remain within the set of feasible solutions Γ. It can be expressed as

$$\operatorname{proj}_\Gamma \mathbf{c}^{(l)} = \mathbf{c}^{(l)} - D^*(D\mathbf{c}^{(l)} - \operatorname{proj}_{\text{time}}(D\mathbf{c}^{(l)})), \tag{4}$$

where $\operatorname{proj}_{\text{time}}$ is the projection operator in time domain defined element-wise as

$$(\operatorname{proj}_{\text{time}}(\mathbf{x}))_n = \begin{cases} y_n, & \text{for } n \in R, \\ \max(\theta, x_n), & \text{for } n \in H, \\ \min(-\theta, x_n), & \text{for } n \in L. \end{cases} \tag{5}$$

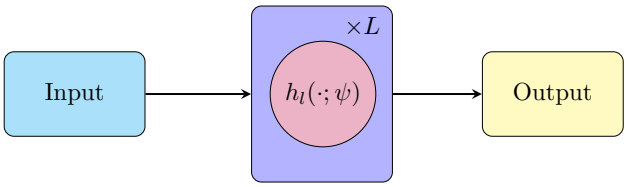

Fig. 1. DRA scheme for nonlearned variant and for tied unfolded algorithm, where ψ are shared between the layers.

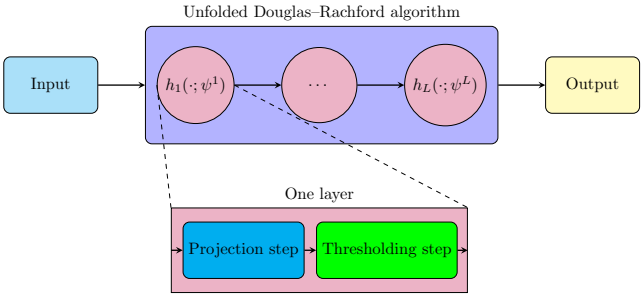

Fig. 2. Unfolded untied DRA with the detail of the first layer/iteration. The projection and thresholding step corresponds to lines 2 and 3 in Alg. 1.

The **thresholding step** (line 3) applies a soft-thresholding operator to promote sparsity in the DGT coefficients:

$$\operatorname{soft}_{\gamma\mathbf{w}}(\mathbf{c}) = \operatorname{sgn}(\mathbf{c}) \odot \max(|\mathbf{c}| - \gamma\mathbf{w}, 0), \tag{6}$$

where $\odot$ is the Hadamard product. We can then write one iteration as $\mathbf{c}^{(l+1)} = h_l(\mathbf{c}^{(l)}, \psi)$, where ψ is the full set of adjustable parameters $(\gamma, \mathbf{w})$ shared across the layers. This is shown in Fig. 1.

C. Deep Unfolding

Despite the good objective results of the DRA, it has two major drawbacks. It is quite difficult to set parameters ψ to achieve a good quality restoration, and a relatively high number of iterations are needed to converge. The deep unfolding aims to solve these two problems. It simplifies the user's choice by optimizing these parameters based on the training data. It is reached by transforming each iteration into a single neural network layer. Formally, we can see unfolding as a composition of functions

$$\mathbf{c}^{(L+1)} = h_L \circ h_{L-1} \circ \cdots \circ h_1(\mathbf{c}^{(1)}), \tag{7a}$$

$$\hat{\mathbf{x}} = D\hat{\mathbf{c}}^{(L+1)} = D(\operatorname{proj}_\Gamma \mathbf{c}^{(L+1)}). \tag{7b}$$

The parameters that are optimized are highlighted in red in Alg. 1. Based on the design choice, we distinguish between *tied* unrolling, with shared parameters along layers shown in Fig. 1, and *untied* unrolling, where the parameters of each layer are optimized separately, illustrated in Fig. 2 [18]. These design choices are explained more in Sec. III

III. Experiments and Evaluation

A. Network architecture

We selected a limited number of layers to guarantee fast inference, which is the main strength of the DU algorithm. Based on the survey paper [7], $L = 15$ layers were chosen. The setting of DGT and iDGT is the same as in [2]: FFT length, $n_{\text{fft}} = 8192$, hop size $a = 2048$, Hann window with size $m = 8192$, and $\mathbf{c}^{(1)}$ is always initialized with zeros.

B. Ablation study of learnable parameters

Firstly, the selection of learnable parameters was examined before training on a larger dataset. A mono segment, approximately 6 seconds in length at 44.1 kHz, was chosen from the MusicNet dataset train set [22]. This dataset consists of various classical music ensembles. The value of ΔSDR (defined in Sec. III-D) was examined, and the variant of weights with the highest value was selected.

The parameter λ was fixed to 1 for all experiments to ensure algorithm stability. The parameters γ and $\mathbf{w}$ in (6) were examined, with the best initialization or system for computing weights being tested. If γ is optimized, then $\mathbf{w}$ is fixed, and vice versa. For all experiments the untied version of DU was used [7].

In the first set of experiments, γ was optimized with a fixed weighting vector $\mathbf{w}$. The vector $\mathbf{w}$ was initially set to all-ones and a parabola-based approach from [2] was applied. However, these experiments did not lead to a significant improvement against the baseline because optimizing only γ is not expressive enough.

With $\gamma = 1$ fixed, the frequency weights $\mathbf{w}$ of coefficients were optimized. Following the symmetry of the FFT, only $n_{\text{fft}}/2+1$ weights were optimized in one layer, then replicated across the other half of the spectrum. This approach, however, did not produce good results, as it was highly sensitive to each frequency bin based on the training data.

To address this, the number of weighting bins was reduced using critical bands, as introduced by Zwicker [23]. With the simplified formulation [24], linearly spaced FFT bins are mapped to 26 bins:

$$\text{Bark}(f) = \left\lfloor \frac{26.81 f}{1960 + f} - 0.53 \right\rfloor . \tag{8}$$

This mapping implies that each Bark bin influences multiple FFT bins based on the human auditory system. The optimized Bark weights are then mapped back to the FFT bins. This approach resulted in the highest ΔSDR value in this single-signal study and was used in the proposed trained network.

C. Training

The 15-layer network was trained on the entire training set of MusicNet with a batch size of 4 and a segment length of 6 seconds for 50,000 iterations. The input signal-to-distortion ratio (SDR) for signal distortion was randomly drawn from 1, 3, 5, 7, and 10 dB for each batch. The network was optimized using Adam [25] with a learning rate of $2 \cdot 10^{-4}$. The selected loss function was error-to-signal ratio (ESR) with high-pass pre-emphasis filter as proposed in [26], to assign higher weight to high-frequency content. The trained weights are shown in Fig. 3. As seen in the figure, the mean of the weights for each bin resembles a parabola shape, which was also shown to be successful in [2]. The low values in the last critical bands (high frequencies) do not follow the parabolic shape, likely because these bands typically contain little signal content. As a result, the optimizer may assign smaller weights to these regions, as they have less perceptual impact on the overall signal.

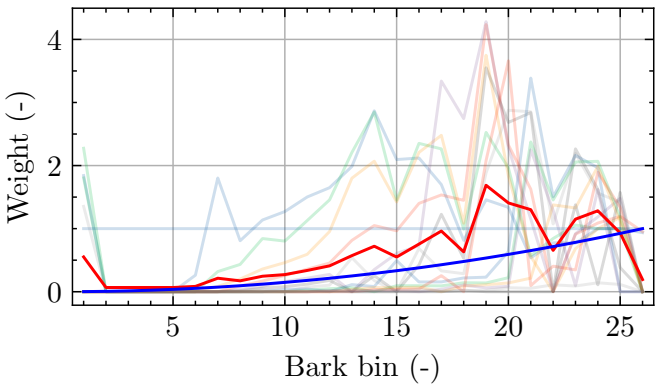

Fig. 3. Learned weights of uDRA in critical bands. The thick red line represents the mean of all layers at a given bin.

D. Comparison with baselines

We compare our learned algorithm uDRA with unweighted DRA and parabola-weighted DRA [2]. To compare the performance, nonlearned algorithms were run with 1000 iterations, as a region where the algorithm should converge and with 20 iterations to compare performance with similar processing time as the learned algorithm.

For evaluation, 10 segments were chosen from the test set of MusicNet. Objective metric ΔSDR is used.

The ΔSDR defines an improvement of SDR between restored signal $\hat{\mathbf{x}}$ and distorted signal $\mathbf{y}$. The SDR for two signals is defined $\text{SDR}(\mathbf{u}, \mathbf{v}) = 10 \log_{10} \frac{\|\mathbf{u}\|_2^2}{\|\mathbf{u}-\mathbf{v}\|_2^2}$, then $\Delta\text{SDR} = \text{SDR}(\mathbf{x}, \hat{\mathbf{x}}) - \text{SDR}(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$ is true target, $\hat{\mathbf{x}}$ is reconstructed and $\mathbf{y}$ is clipped signal [2]. The results are presented in 4. We can see that our algorithm outperforms classical algorithms if the number of iterations is similar, but it does not outperform the algorithm if the number of iterations reach the region of convergence.

E. Computational demands

The processing time of the algorithm is one of the strengths of uDRA. Table I shows the mean and variance of processing times for the algorithms computed on an Nvidia Geforce 4090 GPU and an i7-12700k CPU. The evaluation was performed on the same set of 60 signals used for the objective evaluation. As seen in the table, the inference times for 20 iterations of non-learned DRA and the 15-layer uDRA are quite similar for both the GPU and CPU, but objective results are better for the learned uDRA as presented in III-D.

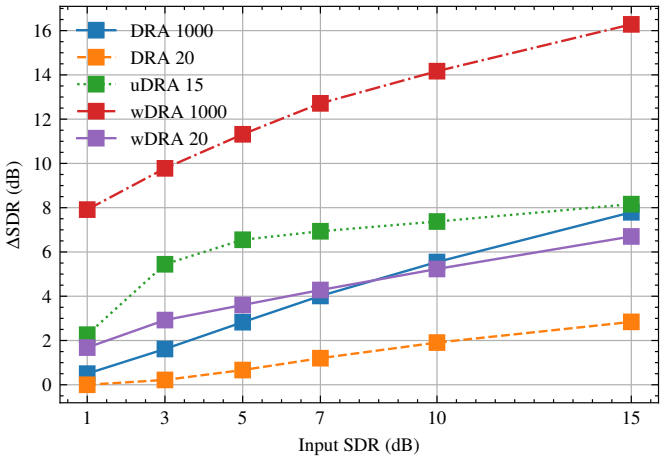

Fig. 4. The mean values of ΔSDR for tested algorithms on 10 signal segments

TABLE I
PROCESSING TIME OF THE ALGORITHMS IN SECONDS

Method	GPU	CPU
DRA 1000	2.41 ± 0.03	252.13 ± 56.58
wDRA 1000	2.44 ± 0.02	251.52 ± 67.83
DRA 20	0.10 ± 0.02	5.05 ± 0.06
wDRA 20	0.04 ± 0.01	5.09 ± 0.04
uDRA 15	0.12 ± 0.03	3.85 ± 0.03

F. Discussion

While the 15-layer uDRA network shows promising results, the current configuration with only 390 parameters may not be sufficient to fully capture the complexity of signal distortion. Although the model performs well compared to classical algorithms, increasing the parameter space could offer more control over behavior of the system.

IV. Conclusion

In this article the deep unfolding of Douglas–Rachford algorithm was applied for audio declipping. The method has shown good performance in terms of objective metric. It beats the classical algorithms in a comparable number of iterations but it has not reached the perceptual quality of the parabola-weighted DRA with the guaranteed number of steps for convergence. However, we hope that a more complex neural network which would replace the one of the steps will solve this problem.

References

[1] P. Záviška, P. Rajmic, A. Ozerov, and L. Rencker, "A survey and an extensive evaluation of popular audio declipping methods," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 1, pp. 5–24, 2021.

[2] P. Záviška, P. Rajmic, and J. Schimmel, "Psychoacoustically motivated audio declipping based on weighted l1 minimization," in *2019 42nd International Conference on Telecommunications and Signal Processing (TSP)*, (Budapest, Hungary), pp. 338–342, July 2019.

[3] B. Li, L. Rencker, J. Dong, Y. Luo, M. D. Plumbley, and W. Wang, "Sparse analysis model based dictionary learning for signal declipping," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 1, pp. 25–36, 2021.

[4] S. Kitić, L. Jacques, N. Madhu, M. 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*, pp. 5939–5943, May 2013.

[5] P. Záviška, P. Rajmic, O. Mokrý, and Z. Průša, "A proper version of synthesis-based sparse audio declipper," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (Brighton, United Kingdom), pp. 591–595, May 2019.

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

[7] E. Chen, X. Chen, A. Maleki, and S. Jalali, "Comprehensive examination of unrolled networks for solving linear inverse problems," 2025.

[8] E. Moliner, J. Lehtinen, and V. Välimäki, "Solving audio inverse problems with a diffusion model," in *2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, jun 2023.

[9] J.-M. Lemercier, J. Richter, S. Welker, E. Moliner, V. Välimäki, and T. Gerkmann, "Diffusion models for audio restoration: A review," *IEEE Signal Processing Magazine*, vol. 41, no. 6, pp. 72–84, 2024.

[10] M. Švento, E. Moliner, L. Juvela, A. Wright, and V. Välimäki, "Estimation and restoration of unknown nonlinear distortion using diffusion," 2025.

[11] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 18–44, 2021.

[12] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (red)," *SIAM Journal on Imaging Sciences*, vol. 10, no. 4, pp. 1804–1844, 2017.

[13] T. Tanaka, K. Yatabe, M. Yasuda, and Y. Oikawa, "APPLADE: Adjustable plug-and-play audio declipper combining DNN with sparse optimization," in *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, may 2022.

[14] X. Wang and S. H. Chan, "Parameter-free plug-and-play ADMM for image restoration," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, Mar. 2017.

[15] K. Zhang, L. Van Gool, and R. Timofte, "Deep unfolding network for image super-resolution," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3217–3226, 2020.

[16] X. Wei, H. van Gorp, L. Gonzalez-Carabarin, D. Freedman, Y. C. Eldar, and R. J. G. van Sloun, "Deep unfolding with normalizing flow priors for inverse problems," *IEEE Transactions on Signal Processing*, vol. 70, pp. 2962–2971, 2022.

[17] Z.-X. Cui, Q. Zhu, J. Cheng, B. Zhang, and D. Liang, "Deep unfolding as iterative regularization for imaging inverse problems," *Inverse Problems*, vol. 40, p. 025011, jan 2024.

[18] P.-H. Vial, P. Magron, T. Oberlin, and C. Fevotte, "Learning the proximity operator in unfolded ADMM for phase retrieval," *IEEE Signal Processing Letters*, vol. 29, pp. 1619–1623, 2022.

[19] S. Wisdom, J. Hershey, J. Le Roux, and S. Watanabe, "Deep unfolding for multichannel source separation," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 121–125, 2016.

[20] W. Yuan, S. Wang, J. Wang, M. Unoki, and W. Wang, "Unsupervised deep unfolded representation learning for singing voice separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 3206–3220, 2023.

[21] P. Combettes and J. Pesquet, "A Douglas–Rachford splitting approach to nonsmooth convex variational signal recovery," *IEEE Journal of Selected Topics in Signal Processing*, vol. 1, no. 4, pp. 564–574, 2007.

[22] J. Thickstun, Z. Harchaoui, and S. M. Kakade, "Learning features of music from scratch," in *International Conference on Learning Representations (ICLR)*, 2017.

[23] E. Zwicker, "Subdivision of audible frequency range into critical bands (frequenzgruppen)," *JOURNAL OF THE ACOUSTICAL SOCIETY OF AMERICA*, vol. 33, no. 2, pp. 248–&, 1961.

[24] H. Traunmüller, "Analytical expressions for the tonotopic sensory scale," *The Journal of the Acoustical Society of America*, vol. 88, p. 97–100, July 1990.

[25] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.

[26] E.-P. Damskägg, L. Juvela, E. Thuillier, and V. Välimäki, "Deep learning for tube amplifier emulation," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 471–475, 2019.