Multimedia Tools and Applications (2021) 80:11903–11915
https://doi.org/10.1007/s11042-020-10197-w

Audio dynamics automatic equalization inspired by visual perception

Luca A. Ludovico[1] · Giorgio Presti[1] · Alessandro Rizzi[1]

Received: 28 December 2019 / Revised: 9 October 2020 / Accepted: 24 November 2020 /
Published online: 7 January 2021

Abstract
This paper explores the behavior of an algorithm called *Audio Dynamics Automatic Equalization* (ADAE). This algorithm has been inspired by research carried out in the context of image restoration: it is the adaptation of a contrast and color unsupervised equalizer for images, called *Automatic Color Equalization* (ACE), into the audio domain. Beside testing if the domain shift from image to audio processing can bring some interesting result, this work also investigates if ADAE behaves like already-known technologies for audio manipulation and restoration. To this end, after a description of the original and the derived algorithms, quantitative test are carried out using typical analyses from the Sound and Music Computing literature, such as frequency response, transfer function, and harmonic distortion. Finally, the paper discusses how the algorithm introduces dynamic range adjustments and non-linear distortions, thus behaving like a dynamics processor, a harmonic exciter, and a waveshaper.

Keywords Audio processing · Dynamics processors · Automatic equalization

1 Introduction

Human sensing is usually a differential process. In this way, our brain is able to adjust automatically to highly varying conditions and to “extract” the meaningful part from the huge amount of information that normally is detected by our senses. From this consideration comes the idea of cross fertilization here presented.

One of the authors has been working for many years on the development and test of image enhancement algorithms, inspired by the characteristics of the *Human Vision System* (HVS). The idea beyond this approach is to mimic the robust and unsupervised capabilities of HVS to adapt to the highly variable configuration of luminance of a real scene. In fact, HVS is able to face high variabilities in the color of the illuminant, of the dynamic range,

✉ Luca A. Ludovico
ludovico@di.unimi.it

[1] Computer Science Department, Università degli Studi di Milano, Via G. Celoria 18, 20133 Milan, Italy

and of the visual contrast of a scene, maximizing the amount of visual information efficiently perceived in every possible condition. A family of algorithms named *Spatial Color Algorithms* (SCA) [14] has been developed mimicking these properties.

Considering that all human senses share a differential and adaptive nature, we have decided to apply ACE, an image enhancer algorithm of the SCA family, to a sound stream. ACE, standing for *Automatic Color Equalization*, performs a local, differential and non-linear computation of local channel lightness, followed by an available dynamic range maximization [2, 11–13]. It will be described in more details in the next section.

The most relevant properties of ACE are the following: it does not require any user supervision, and it adjust its filtering effect on the image content, performing a local and variable lightness and contrast adjustment. In other words, if the input image presents balanced colors and contrast and a full dynamic range, ACE leaves it unmodified; conversely, it can perform a strong filtering if color, contrast or dynamic range of the input image requires a massive adjustment.

This property of unsupervised, self-tuning, local adjustment can be very useful also in the field of audio and music processing. For this reason, we will apply the ACE approach to a different context, namely the audio domain, in order to test its possible uses to restore information.

Among many instruments used in audio production and restoration industry, it is worth introducing some tools that are particularly relevant in this context, that are *Dynamics Processors* (DPs), *Waveshapers* (WSs) and *Harmonic Exciters* (HEs). They are all non-linear processes; HEs and WSs are time-invariant, or with a memory in the order of a few milliseconds, while DPs are characterized by a longer memory, in the order of some seconds.

DPs are a family of processors that encompasses tools like *compressors*, *expanders*, *limiters*, and *gates*. Their goal is to alter signal level (i.e. the volume) according to a *ratio* parameter only when the input level is above (*compressor* and *limiter)* or below (*expander* and *gate*) a *threshold* value. To prevent waveform distortion, the effect of a DP is introduced and released gradually according to parameters called *attack time* and *release time*. Overall gain change is compensated by a *make-up gain*, that is a linear gain change applied to the output signal, in order to bring the new peak level to the input peak level [6, 8]. DPs are used to sculpt the dynamic level of a signal, e.g. for reducing level differences of a vocal track, or for controlling the transient phase of plucked and percussive instruments. This kind of processors is useful in hearing aid devices, too.

HEs are processors that introduce subtle non-linearities (hence harmonic distortion) with the goal of enriching the spectral content of a sound, e.g. to add more clarity to a vocal track, or to brighten a piano recording [15].

In music production, also heavy non-linearities are used to deliberately mangle the sound. This is the goal of WSs, which may be significantly different from each other.

DPs, WSs, and HEs can be realized in many ways, with a variety of possible implementations, still being invented and patented by the audio technology industry. A *best tool*, adequate for all circumstances, does not exist, rather the choice depends on users' needs and preferences.

From a signal-processing perspective, non-linear distortion and dynamic range reduction are considered as an objective degradation in signal quality, so they should be avoided. Conversely, in the audio production context, objective degradation can be performed so as to obtain a subjective improvement based on aesthetics and cultural practices [9]. A clarifying example is offered by distorted instruments: they would not be tolerated in classical

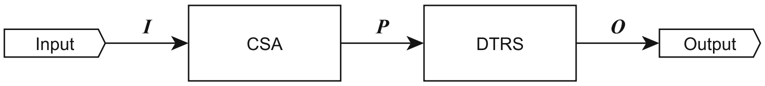

Fig. 1 The two stages of ACE: *Chromatic Spatial Adjustment* (CSA) and *Dynamic Tone Reproduction Scaling* (DTRS). CSA is responsible for the actual processing of the input I, and DTRS maps the dynamic range of P to that of the output encoding format

music recordings, whereas distortion can be a timbral distinguishing feature for a rock performance.

2 The ACE algorithm in the visual domain

In order to show how ACE works in the visual domain, that is the context it was conceived for, we will first present its formulas and then discuss some results.

ACE is composed of two stages, as shown in Fig. 1: during the first stage, it performs pixel adjustments, depending on chromatic and spatial properties of the input image I. This step generates an intermediate image P, that can be considered as a sort of perceived image. In order to make the image displayable, the second stage uses estimated white and gray to map the dynamic range of P into the available one. Each color channel is processed separately, a characteristic of this family of algorithm [14].

Each pixel is computed as follows. Let us fix the coordinates i of a target pixel and let j denote the generic pixel in the rest of the image. Given a constant $\alpha > 1$, with $\alpha \in \mathbb{R}$, we construct the odd function $\mathrm{s}_\alpha : [-1, 1] \to [-1, 1]$, defined as:

$$\mathrm{s}_\alpha(t) = \begin{cases} -1 \text{ if } -1 \leq t \leq -\frac{1}{\alpha} \\ \alpha t \text{ if } -\frac{1}{\alpha} < t < \frac{1}{\alpha} \\ +1 \text{ if } \frac{1}{\alpha} \leq t \leq 1 \end{cases} \tag{1}$$

where α and s_α are called, respectively, *slope* and *slope function*.

In the next discussion, we will address the analysis of s_α when t assumes the values $t = I(i) - I(j)$, i.e. we will deal with $\mathrm{s}_\alpha(I(i) - I(j))$, with $I(i)$ constant (the pixel to compute) and $I(j)$ variable (since j runs over the whole image).

The reason for the use of differences in ACE is that they are an easy way to implement the differential nature of spatial comparisons.

We also consider a *weight function* $w_i : \mathfrak{I} \times \mathfrak{I} \to (0, 1)$, where $w(i, j)$ is a monotonically decreasing function representing the induction weight, i.e. the mutual chromatic influence between pixels i and j in $\mathfrak{I}$ [3, 4, 7, 18]. The tuning performed in [12] and [13] showed that a parameter-free weight function that corresponds to overall good performances is

$$w(i, j) = \frac{1}{d(i, j)}, j \in \mathfrak{I} \setminus \{i\}, \tag{2}$$

being d the Euclidean distance between pixels.

The lightness computation of i is performed by ACE in two steps, as shown in Fig. 1. First of all we compute the *chromatic spatial adjustment*

$$P(i) = \frac{\sum_{\mathfrak{I}\setminus\{i\}} \mathrm{s}_\alpha(I(i) - I(j))w(i, j)}{\sum_{\mathfrak{I}\setminus\{i\}} w(i, j)}, \tag{3}$$

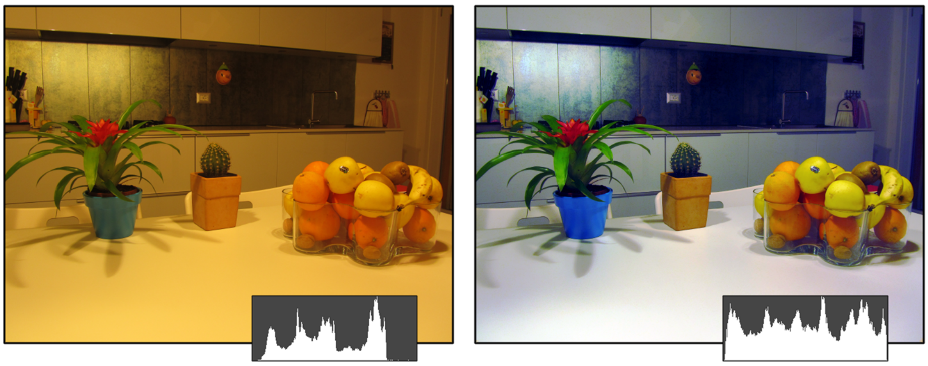

Fig. 2 An example of ACE at work. The input image on the left is processed with a slope value of 20. Result is shown on the right. Image histograms are visible under the pictures

where the division by $\sum_{\Im \setminus \{i\}} w(i, j)$ is needed to correctly normalize the computation near the edges [12]. Then, set $M = \max_{i \in \Im}\{P(i)\}$, we perform a *dynamic tone reproduction scaling*, defining the lightness of the target i computed by ACE as

$$L^{\mathrm{ACE}}(i) = \frac{1}{2} + \frac{P(i)}{2M}. \tag{4}$$

The main effect of ACE is an unsupervised contrast and color adjustment and a consequent maximization of the available dynamics. These two effects are closely connected. The distance weight function allows ACE to perform a local contrast enhancement that is fundamental for the differential nature of human vision, and this is done without any a-priori information about the original contrast in the image to filter. In the same way, if a color dominant is present, it is usually removed, as an effect of the independent channel local filtering. All these properties are visible in the filtering example shown in Fig. 2.

3 The ACE algorithm in the audio domain

We refer to the ACE transformation in the audio domain as ADAE, standing for *Audio Dynamics Automatic Equalization*. Adapting the ACE approach to the audio waveform requires a domain shift. In this sense, we have to highlight and exploit some similarities in the way images and sounds are represented in the digital domain.

The input values for the algorithm correspond no longer to the intensity of pixels, but to the amplitude of audio samples. Consequently, the audio signal presents a reduced dimensionality with respect to digital images: the input changes from a multi-dimensional spatial domain to a mono-dimensional time domain.

A first adaptation regards the calculation of distance, which – for two samples of a signal S_c at time i and j – can be simplified as $abs\,(i - j)$. The amplitude differences of samples are thus weighted by their temporal distance instead of spatial distance. The ADAE formula for the first stage becomes:

$$P_c(i) = \sum_{j \in S,\, j \neq i} \frac{s_\alpha(S_c(i) - S_c(j))}{abs(i - j)}, \tag{5}$$

where P_c is the intermediate signal, S_c is the input signal, the subscript c identifies the processed channel, $S_c(i) - S_c(j)$ provides the difference between samples at time i and j, and $s_\alpha()$ is the same function introduced in Section 2 for the image domain.

Another major difference due to the domain shift is the meaning of input values for the *scaling* stage of (4), which is strongly related to the domain of the processed information. In particular, the sums calculated during the first stage must be scaled to the new output co-domain $[-1 \cdots +1]$. The default method is peak normalization, that optimizes the available dynamic range in output. Please note that, similarly to ACE in the visual domain, other methods can be chosen depending on the expected results. For example, RMS scaling [1] can be used to bring the output signal at the same RMS of the input, but it does not ensure that the signal will not clip once saved.

Finally, it is worth underlining another similitude between ACE and ADAE: multichannel audio is handled by processing each channel separately, like the algorithm manages multichannel RGB images in the visual domain.

In conclusion, a model originally conceived for human vision has been applied to sound perception, keeping the same approach towards information computing.

3.1 Optimization of the algorithm

The reduced dimensionality of audio signals, with respect to the image domain, does not necessarily imply a reduction of the complexity of the algorithm (since an audio signal can be seen as a linearized image).

However, a reduction in complexity can be achieved by other means. Given N as the number of pixels or audio samples, the plain version of ACE has a complexity of $\mathcal{O}(N^2)$. Images are not time dependent, so all N pixels must be considered for the computation of each point of the output. Conversely, in the audio domain two samples distant in time hardly influence the mutual perception, which enables the ability to work *locally*. This reduces the overall number of operations by limiting the *maximum distance* between samples to be compared. Basically, the computation occurs inside a sliding window of size M, thus lowering the complexity to $\mathcal{O}(N \cdot M)$. This can collapse to $\mathcal{O}(N)$ when $M \ll N$ and can reach $\mathcal{O}(N^2)$ when $M \approx N$.

The definition of the window size depends on which aspect of sound is going to be addressed, and becomes a new parameter of the algorithm. Different window sizes produce effects that cope with adaptation and/or adjustment phenomena, such as temporal masking in case of small windows (say less than half a second) and loudness adaptation for wider windows. Moreover, this temporal window may be asymmetric due to the analyzed sample: pre-sample distance – called *look-behind* – and post-sample distance – called *look-ahead* – may be different. This difference between pre- and post-samples may also justify the use of different α for the look-ahead and look-behind portion of the signal. In next sections, an exploration of these parameters will be discussed.

Finally, it is worth citing that the behavior at the signal boundaries can be handled in very different ways. The prototype software zero-pads the empty part of the sliding window.

[1]RMS stands for root mean square. RMS normalization is equivalent to L2 normalization.

4 Test signals

In order to assess the effects introduced by ADAE, we carried out quantitative tests and parameter exploration aimed at measuring objective signal changes and identify possible uses of ADAE.

To this goal, we processed through ADAE two synthesized signals, created with the aim of measuring the non-linear effects introduced in time and frequency domains. Such signals are: i) a logarithmic sinusoidal sweep, and ii) a train of sinusoidal bursts with different amplitudes.

In the former case, we adopted a technique based on the simultaneous measurement of impulse response and distortion with a logarithmic sweep signal. This representation is referred to as the *Hammerstein system model* (or *Diagonal Volterra Kernels*) [1, 10], which can be considered as an extension to the non-linear domain of the *impulse response* technique, providing an exhaustive description of system behavior, including short-term memory effects. The measurement technique consists in analyzing a non-linear system by feeding a logarithmic-frequency sinusoidal sweep in input, and then convolving the output with the corresponding inverse sweep. This generates a new signal which is composed by a train of impulses, each one corresponding to the response of different orders of distortion (i.e. the different harmonics) plus a final impulse carrying the response of the linear component [5]. This technique is frequently used to model non-linear systems in the audio processing literature [16, 17]. The used sweep had a frequency span from 10 to 22000 Hz. An adequate oversampling has been used to reduce the aliasing effect introduced when processing higher frequencies. As an example, Fig. 3 shows the spectra of the impulses mentioned above, coming from the response of a HE (on the left) and a more aggressive WS, namely an *overdrive distortion* (on the right). The image highlights how the tested HE has a non-flat linear frequency response (see the bump around 100 Hz of the dark line marked as *linear* in the legend), and adds all harmonics above 100 Hz, while the analyzed WS has a more flat response, and adds only odd harmonics (i.e. only those marked as *3rd*, *5th*, *7th order etc.* in the legend) uniformly allover the spectrum.

In the latter case, namely for the train of sinusoidal bursts, we wanted to grasp broadband envelope dynamics modifications, by means of simple observations of the time-domain transfer function of the algorithm. The transfer function is derived from the comparison of

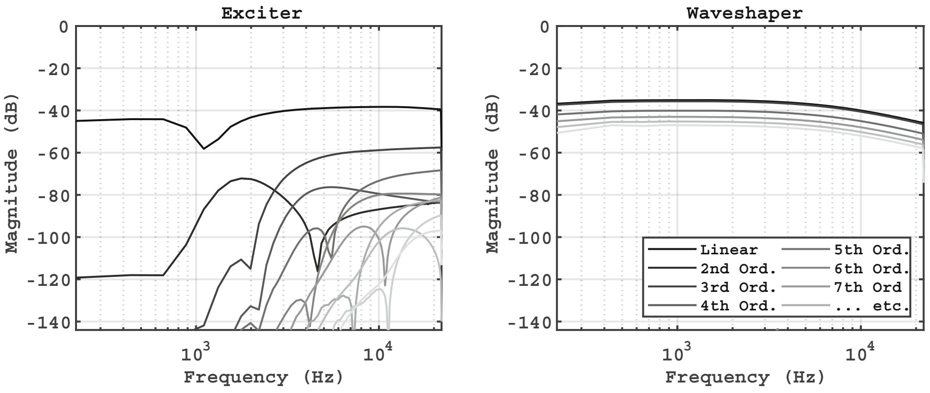

Fig. 3 Hammerstein models of a HE (an emulation of an Aphex Systems LTD. Aural Exciter, on the left) and a WS (a generic overdrive distortion, on the right)

the amplitude envelope of bursts signals before and after ADAE processing. Burst signals are generated by modulating the amplitude of a 5 kHz sine wave, and output envelopes are extracted as the magnitude of the analytical signal (i.e. the Hilbert transform) of the processed sounds.

The results of this analysis in the audio domain usually show a curve characterised by two different linear phases, with different slopes. If the system is a DP, slopes are a function of the DP *ratio*; the knee between the slopes depends on the *threshold*, and long-term memory effects, visible by comparing the input and output envelopes against time, are function of the *attack* and *release* time. An example can be seen in Fig. 4, where the behavior of a compressor is shown: the offset of the linear phase of the curve, compared to the unity gain (represented by the dashed line), can be due to an input gain of about 10 dB, or a makeup gain of the same amount, depending on the implementation. In the first case the threshold is at 0 dB_{fs} (when the curve bends toward a horizontal slope), while, in the second case, the threshold is at -10 dB_{fs} approximately. In both cases, the horizontal slope of the second phase of the curve indicates a very high compression ratio. The fact that the transition is either smooth or sudden depends on the implementation of the DP.

In the right plot, it is clear how the original dynamic range of 40 dB is reduced to about 30 dB, except for the attack and release phases, lasting about 250 ms each. Further detail on how to read these plots can be found in [6, 8].

For both test signals, namely the logarithmic sinusoidal sweep and the train of sinusoidal bursts, in our analysis different values of ADAE parameters are explored, trying to figure out how such parameters affect the output signal.

5 Remarks

5.1 Spectral effects

The ADAE algorithm can be briefly described as the amplification and saturation of the signal in a “differential” domain, followed by a cumulative sum operation. This approximation suggests an offset-removal behavior, a “differences enhancer”, and an amount of distortion which is proportional to the input frequency. This is confirmed by the impulse response of the different orders of distortion shown in Fig. 5.

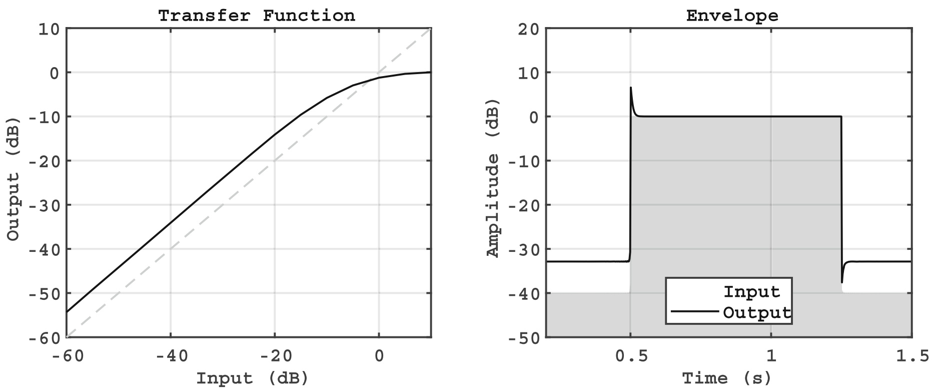

Fig. 4 Transfer function (on the left) and time domain behavior (on the right) of a common compressor

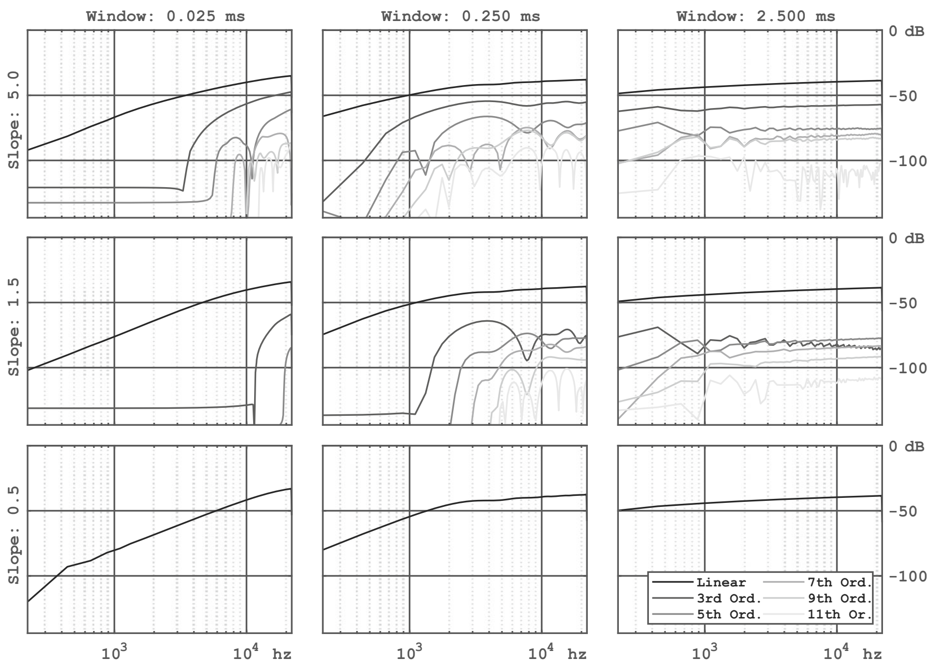

Fig. 5 System frequency response when varying ADAE parameters; different window sizes are represented in columns, and slope values in rows

In particular, the amount of distortion is proportional to slope α, which, at the same time, controls the frequency above which ADAE starts adding harmonics. The abrupt transition is due to the s_α function (1), which shows a sudden change from a linear to a non-linear behavior, as clearly illustrated by Fig. 6.

Across the rows of Fig. 5, it can also be noted that an increase of window size corresponds to a lowering of the cut-off frequency, and that even window size has an effect on the frequency above which ADAE starts to add harmonics. The two aspects are clearly linked: by lowering the volume of low-frequencies, it is harder for them to hit the non-linear part of s_α.

Finally, despite the temporal asymmetry of look-ahead and look-behind parameters, only odd harmonics are produced, since s_α is an odd function.

This behavior can be assimilated to the one of both a HE and a WS, depending on the parameters' values. In particular, window sizes smaller than 1 s and slopes between 1 and 5 s may result in a HE-like effect, while larger slopes and window sizes result in an overdrive-like effect.

5.2 Dynamics and time-domain effects

Given a slope greater than 1, for small values of window size (e.g., less than 20 ms in total), ADAE only manifests a local effect, namely a non-linear distortion. Changes in short-term dynamics appear for medium values of the temporal parameters (20 ms to 5 s), while higher values also influence long-term level changes, thus making ADAE fall also into the DPs category. However, it is worth underlining that the local effects are maintained also when

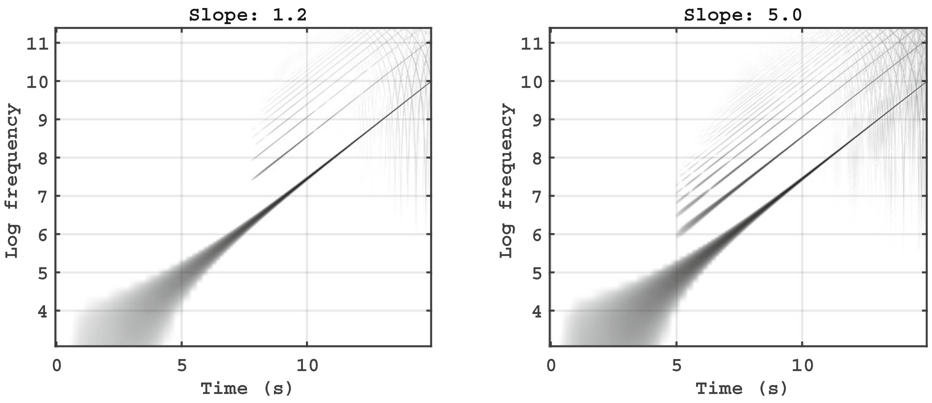

Fig. 6 Spectrogram of the sinusoidal logarithmic sweep processed with different slope values

enlarging the window size. This aspect can be annoying, since local effects – even when pleasant to the ear – consist in not always desirable audio distortion. In other words, the HE/WS behavior cannot be decoupled from the DP behavior.

Figure 7 presents the effect that ADAE has on the signal envelope in the time domain, revealing how ADAE recalls a compressor. In particular, s_α determines the compression ratio, thus it cannot be controlled by changing ADAE parameters. The slope controls both the input gain and the threshold: in the case depicted in Fig. 7, a slope of 10 means that the input of s_α is multiplied by 10 (i.e. +20 dB), and all resulting values above 0 dB_{fs} are then truncated by s_α, thus the threshold can be traced back at −20 dB_{fs} of the input scale. Makeup gain is realized by the post-processing normalization phase of ADAE.

The main difference from a traditional compressor shows up in the interpretation of time constants. As highlighted on the right side of Fig. 7, look-behind time can be considered as the attack and release time of traditional compressors (both set to the same value), while look-ahead time can be seen as the same look-ahead parameter featured by some advanced compressors, namely a slow gain change anticipating signal peaks. This behavior may be

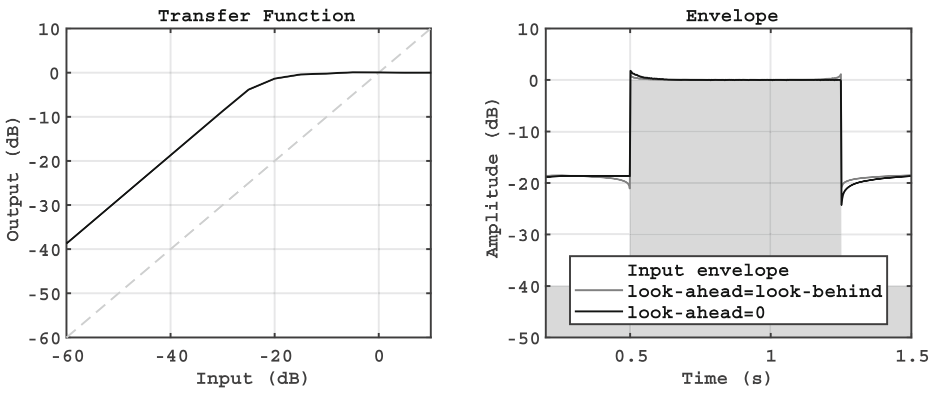

Fig. 7 Transfer function (on the left) and time domain behavior (on the right) of ADAE, resembling those of a compressor. Slope: 10, look-ahead and look-behind: 0.25

desirable in contexts of dynamics manipulation, but – like we said – preserving this characteristic without introducing audible distortion is not possible without changing the very nature of the ADAE algorithm.

6 Conclusions

In this paper we presented ADAE, an audio algorithm inspired by research carried out in the context of image restoration.

The original research question was to test if the shift from the image to the audio processing domain could bring interesting results. Some relevant differences emerged, first of all the time independence of images against the time dependence of monophonic audio. Consequently, it was necessary to reconsider a number of key concepts of the original formulation, e.g., the distance among pixels vs. samples. We believe in the possibility of reusing the same algorithms over different data types, and with our work we try to encourage the multimedia community to embrace this kind of experiments.

Focusing on sound computing, a secondary goal was to investigate if ADAE behaves like already-known technologies for audio manipulation. To this end, the algorithm was tested on sounds with different characteristics. In general terms, the behavior can be broadly compared to a combination of other audio-processing techniques already in use, such as DPs, HEs and WSs, as demonstrated by the tests.

In conclusion, ADAE combines the characteristics of the three mentioned categories of audio processors, depending on the chosen parameters, and can be employed in a number of cases, ranging from the enhancement of high frequencies (usually in vocal tracks) to instrument distortion and instrument dynamics processing. Unfortunately, unlike ACE, this behavior is not easy to be used in an automated way; rather, it requires to adopt the trial-and-error methodology to find the optimal values for parameters so as to achieve the desired task.

For this reason, future work will mainly focus on perceptual tests aiming to compare ADAE with industry-standard audio processors falling in the categories of DPs, WSs, and HEs. The goal is to measure the perceived quality of the audio produced by the algorithm in specific scenarios, with reference to a variety of already implemented tools. This process is expected to unveil further cases benefiting from ADAE's potential.

In order to facilitate the analysis and testing of ADAE by the scientific community, a MATLAB implementation of the algorithm is presented in Appendix. Some audio examples showing the results of ADAE processing are available at the following URL: https://www.lim.di.unimi.it/demo/adae.php. Specifically, the first test concerns a band-limited music signal, that is poor in high frequencies; the second case involves a band-limited vocal signal with dynamic excursion issues; the third example addresses bass saturation; finally, the fourth test concerns a music signal with reduced dynamics. For all the mentioned cases, both the results of traditional processing and those produced by ADAE are presented in the form of audio examples and properly commented.

Acknowledgments The authors would like to thank Carlo Gatta, a friend and colleague, for the idea that originated the present research activity and Cristian Bonanomi for the picture. This paper is an outcome of the project "Discovering Patterns in Multi-Dimensional Data" financed by the Department of Computer Science, University of Milan.

Appendix: ADAE Source Code

```
function [y] = ADAE(x, sr, span, slope)
% [y] = ADAE(x, sr, span, slope)
%   span can be a scalar or [lookBehind, lookAhead] (seconds
    )
%   slope can be a scalar or [behindSlope, aheadSlope]

    wb = waitbar(0,'Initializing...');

    if numel(span) == 1
        pre = ceil(span * sr);
        pos = ceil(span * sr);
    else
        pre = ceil(span(1) * sr);
        pos = ceil(span(2) * sr);
    end

    if numel(slope) == 1
        bslope = slope;
        aslope = slope;
    else
        bslope = slope(1);
        aslope = slope(2);
    end

    l = size(x,1);
    t = (linspace(0,l-1,l)/sr).';
    y = zeros(l,1);
    tic
    for ii = 1:l
        inB  = max( ii-pre ,1);
        outB = max( ii-1   ,1);
        inA  = min( ii+1   ,l);
        outA = min( ii+pos ,l);
        A = 0; B = 0;
        if ii > 1
            B = pci(x(ii),t(ii),x(inB:outB),t(inB:outB),
                bslope);
        end
        if ii < l
            A = pci(x(ii),t(ii),x(inA:outA),t(inA:outA),
                aslope);
        end
        y(ii) = (B./max(pre,1)) + (A./max(pos,1));
        prog = ii/l;
        timeToUpdate = mod(ii,round(30*l/(pre+pos))) == 0;
        if timeToUpdate
```

```
                eta = (toc/ii) * (l-ii);
                waitbar(prog, wb, {'Computing␣ADAE...', ...
                    sprintf('%0.1f%%␣-␣ETA:␣%s', ...
                    prog*100, datestr(seconds(eta), 'HH:MM:SS'))
                        });
            end
        end
        close(wb);
end

function [y] = sa(x,slope)
        y = min(max(x*slope,-1),1);
end

function [y] = pci(si,ii,Sj,Ij,slope)
        y = sum(sa(si-Sj,slope)./abs(ii-Ij));
end
```

Funding Open access funding provided by Universit degli Studi di Milano within the CRUI-CARE Agreement.

References

1. Bedrosian E, Rice SO (1971) The output properties of Volterra systems (nonlinear systems with memory) driven by harmonic and Gaussian inputs. Proc IEEE 59(12):1688–1707
2. Bertalmío M, Caselles V, Provenzi E, Rizzi A (2007) Perceptual color correction through variational techniques. IEEE Trans Image Process 16(4):1058–1072
3. Creutzfeldt O, Lange-Malecki B, Wortmann K (1987) Darkness induction, retinex and cooperative mechanisms in vision. Exp Brain Res 67(2):270–283
4. Creutzfeldt O, Lange-Malecki B, Dreyer E (1990) Chromatic induction and brightness contrast: a relativistic color model. JOSA A 7(9):1644–1653
5. Farina A (2000) Simultaneous measurement of impulse response and distortion with a swept-sine technique. In: Audio engineering society convention, p 108. http://www.aes.org/e-lib/browse.cfm?elib=10211
6. Giannoulis D, Massberg M, Reiss JD (2012) Digital dynamic range compressor design – a tutorial and analysis. J Audio Eng Soc 60(6):399–408
7. Jameson D, Hurvich LM (1964) Theory of brightness and color contrast in human vision. Vis Res 4(1):135–154
8. Jeffs R, Holden S, Bohn D (2005) Dynamics processors—technology & application tips. RaneNotes 155(2005):1–28
9. Katz RA (2003) Mastering audio: the art and the science. Butterworth-Heinemann
10. Kibangou AY, Favier G (2006) Wiener-Hammerstein systems modeling using diagonal Volterra kernels coefficients. IEEE Signal Process Lett 13(6):381–384
11. McCann JJ, Rizzi A (2011) The art and science of HDR imaging, vol. 26. Wiley

12. Rizzi A, Gatta C, Marini D (2003) A new algorithm for unsupervised global and local color correction. Pattern Recogn Lett 24(11):1663–1677
13. Rizzi A, Gatta C, Marini D (2004) From retinex to automatic color equalization: issues in developing a new algorithm for unsupervised color equalization. J Electron Imag 13(1):75–84
14. Rizzi A, McCann JJ (2007) On the behavior of spatial models of color. In: Eschbach R, Marcu GG (eds) Color imaging XII: processing, hardcopy, and applications, vol 6493. International society for optics and photonics, SPIE, pp 11–24, https://doi.org/10.1117/12.708905
15. Shekar P, Smith JO III (2013) Modeling the harmonic exciter. In: Audio engineering society convention, pp 135. Audio Engineering Society
16. Tronchin L (2013) The emulation of nonlinear time-invariant audio systems with memory by means of Volterra series. J Audio Eng Soc 60(12):984–996
17. Tronchin L, Coli VL, Gionfalo FF (2017) Modelling nonlinearities on musical instruments by means of Volterra Series. In: Audio engineering society convention, pp 142. Audio Engineering Society
18. Zaidi Q (1999) Color and brightness induction: from Mach bands to three-dimensional configurations. In: Gegenfurtner K, Sharpe L (eds) Color vision: from genes to perception. Cambridge University Press, New York, pp 317-343