A DIRECT MICRODYNAMICS ADJUSTING PROCESSOR WITH MATCHING PARADIGM AND DIFFERENTIABLE IMPLEMENTATION

Shahan Nercessian, Russell McClellan, and Alexey Lukin

iZotope, Inc.
Boston, MA, USA
shahan@izotope.com | rmcclellan@izotope.com | alex@izotope.com

ABSTRACT

In this paper, we propose a new processor capable of directly changing the microdynamics of an audio signal primarily via a single dedicated user-facing parameter. The novelty of our processor is that it has built into it a measure of *relative level*, a short-term signal strength measurement which is robust to changes in signal macrodynamics. Consequent dynamic range processing is signal level-independent in its nature, and attempts to directly alter its observed relative level measurements. The inclusion of such a meter within our proposed processor also gives rise to a natural solution to the dynamics matching problem, where we attempt to transfer the microdynamic characteristics of one audio recording to another by means of estimating appropriate settings for the processor. We suggest a means of providing a reasonable initial guess for processor settings, followed by an efficient iterative algorithm to refine upon our estimates. Additionally, we implement the processor as a differentiable recurrent layer and show its effectiveness when wrapped around a gradient descent optimizer within a deep learning framework. Moreover, we illustrate that the proposed processor has more favorable gradient characteristics relative to a conventional dynamic range compressor. Throughout, we consider extensions of the processor, matching algorithm, and differentiable implementation for the multiband case.

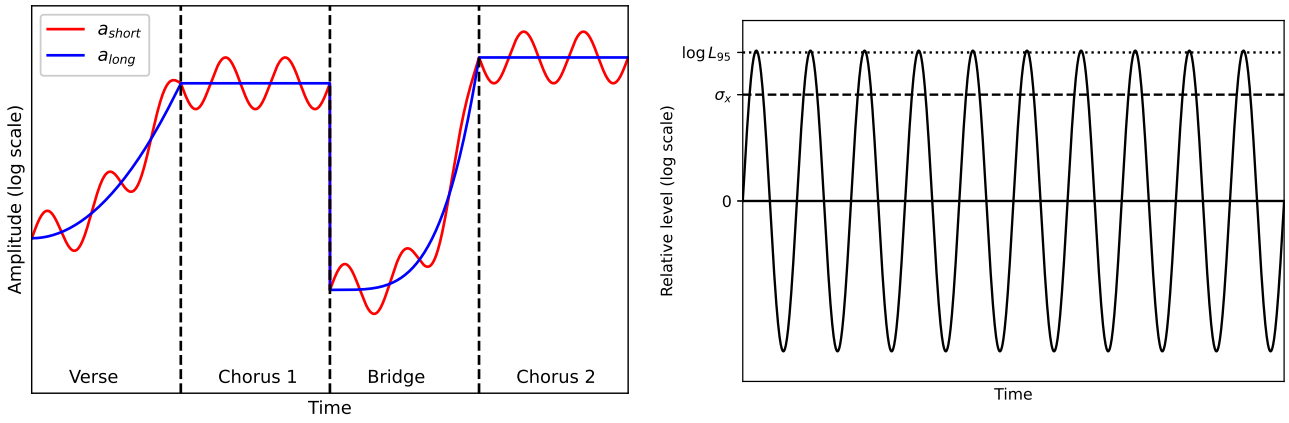
1. INTRODUCTION

The concept of dynamic range in a musical recording refers to its variation of loudness over time. Dynamic range can be divided into *macrodynamics* and *microdynamics* [1]. Macrodynamics refers to the variation of the loudness over a relatively long time scale, such as between the chorus and verse in a song. Variations over a shorter time scale — for example, the variation in loudness between notes in the same musical phrase, or between the attack and sustain sections of the same note — are known as *microdynamics*. We generally perceive macrodynamics and microdynamics differently. Recordings with extremely high levels of macrodynamics will prompt listeners to turn down the volume of loud passages and/or turn up the volume of soft passages in order to maintain a consistent listening level. Meanwhile, recordings with high levels of microdynamics are sometimes described as "punchy" or "snappy" [2], whereas passages with low levels of microdynamics may be considered to be more "sustained," and this phenomenon is generally agnostic to a signal's macrodynamics.

Throughout the various stages of the recording and production process, the microdynamics of a signal can be adjusted through the use of audio processors. Many such dynamic range adjustment processors are in use, including for example, the transient shaper. However, arguably the most popular one is the *dynamic range compressor,* which is commonly used to to reduce the dynamic range of a recording (alongside its complementary processor, the *dynamic range expander,* which can increase dynamic range) [3]. Despite its ubiquitious utilization, the standard compressor/expander paradigm leaves a lot to be desired. Its parameterization is fairly unintuitive, with complex interactions between parameters. Its processing is signal-level dependent, and therefore, the exact behavior of any given setting cannot be intuited until it is placed in the context of the signal it is applying processing to. Moreover, its operation can be affected by the macrodynamics over the course of a piece of music, so that different musical passages undergo different amounts of dynamic range adjustment in the absence of any parameter automation.

In addition to dynamics processing, there has been significant research aimed at objectively characterizing and measuring the amount of microdynamics contained in an audio signal, motivated by a desire to provide actionable metering information to music creators and recording engineers alike [2, 4, 5, 6]. The most primitive of these measures include the well-known crest factor [7], while more perceptually relevant metrics include the Loudness Dynamic Range (LDR) measure [2]. With the advent of intelligent audio systems that leverage computer-aided decision making [8, 9]), matching a specific sonic quality of source material to that of a reference is an increasingly prevalent trend [10]. This has proved to be more tractable for matching overall level (by means of level-matching [5]) or tone (by means equalizer matching [11, 12, 13]), and poses more challenges for non-linear effects such as dynamics processing [14, 15, 16], distortion [17, 18], etc. It seems only natural that such measures of microdynamics would prove useful for creating relevant objectives for matching the microdynamics between source and reference signals, especially if said measures were ingrained into the dynamic range adjusting processors themselves.

In this paper, we propose a new dynamic range processor that can directly alter signal microdynamics, primarily by a single manipulation parameter. Built into our processor, and crucial to its formulation, is a robust measure of signal strength which we directly attempt to alter during processing. We also offer a natural extension of the processor to the multiband case, in which case, the processor contains not only independent processing controls, but also independent meters for each sub-band on which it operates. The formulation of such a processor naturally provides a solution to the task of dynamics matching, in which case we are interested in transferring the microdynamic characteristics of a reference sig-



(a) (b)

Figure 1: *(a) Notional short- and long-term amplitude envelopes over the course of a musical arrangement and (b) corresponding relative level, where horizontal lines for 0,* $\log L_{95}$ *and* σ_x *represent* $a_{long}[n]$*, the LDR, and the standard deviation of* $\log L[n]$*, respectively.*

nal to some source material via the automatic selection of processor parameters. To this end, we offer a solution to the problem by means of an iterative algorithm. We also implement our proposed processor as a differentiable recurrent layer, and highlight the ability of such a layer to be wrapped around gradient descent optimizers whilst exhibiting fruitful gradient characteristics.

The remainder of this paper is organized as follows: In Section 2, we provide a brief overview of relevant measures of microdynamics and the conventional dynamic range compressor. In Section 3, we describe the proposed dynamic range adjusting processor. Section 4 describes an iterative solution to the dynamics matching problem using the proposed processor, while Section 5 involves the differentiable realizations of dynamic range processors with illustrative experiments therein. Lastly, Section 6 draws conclusions and alludes to future work.

2. BACKGROUND INFORMATION

2.1. Objective measures of microdynamics

While many objective measures of microdynamics have been proposed, a measure called LDR correlated best with perceived microdynamics in a perceptual test by Skovenborg [2]. Here, we highlight some of the key steps to its computation. Given a sampled input signal $x[n]$, the LDR begins by defining the ratio between a short-term amplitude envelope $a_{short}[n]$ and a long-term amplitude envelope $a_{long}[n]$ as

$$L[n] = \frac{a_{short}[n]}{a_{long}[n]} = \frac{f_{rms}(x[n]^2; \tau_{short})}{f_{rms}(x[n]^2; \tau_{long})} \tag{1}$$

We will continue to refer to ratios of this form as "relative level" measurements of a signal $x[n]$. The function $f_{rms}(x[n]^2; \tau)$ implements the running root mean square (RMS) computation

$$a^2_{rms}[n] = \alpha x[n]^2 + (1-\alpha) a^2_{rms}[n-1] \tag{2}$$

where the smoothing parameter α for a given time constant τ is generically defined as

$$\alpha = 1 - \exp[-1/(\tau f_s)] \tag{3}$$

and f_s is the audio sampling rate. The LDR uses a "fast" integration time τ_{short} on the order of 50 ms, along with a "slow" integration time τ_{long} on the order of 3 seconds.

The purpose of this relative level metric is to measure changes in signal strength in such a way that it can compensate for the overall level of a particular musical passage, as is notionally illustrated in Figure 1. In this contrived example, the overall signal level (as captured by $a_{long}[n]$) changes from one section of a musical arrangement to another, which can have the effect of distorting a measure of microdynamics defined purely in terms of $a_{short}[n]$. Meanwhile, the relative level can capture the ways in which $a_{short}[n]$ varies *about* $a_{long}[n]$. Finally, the LDR quantifies microdynamics as the high percentile (typically 95th percentile) or maximum value of all observed relative levels (in decibels) over an analysis window. Note that throughout this paper we will use the notion of the log scale and decibels interchangeably for notational convenience, as they are equivalent to each other up to a scaling factor. Accordingly, we denote the LDR as $\log L_{95}$, as annotated in Figure 1b.

LDR is conceptually similar to an earlier Dynamic Spread (DS) measure proposed in [5], in that both involve calculating statistics on a time-varying measure of loudness. Specifically, DS considers the standard deviation and/or mean absolute deviation of its loudness measure instead of the order statistic used by the LDR. An important distinction is that LDR introduces the notion of the long-term level calculation which is absent from the DS measure. The long-term envelope represents signal macrodynamics, thus dividing by this slow envelope causes the LDR to measure *only* the microdynamics of the recording. In contrast, DS includes information about the microdynamics, but it is "contaminated" by the effect of signal macrodynamics.

2.2. Dynamic range compressor

We briefly describe the dynamic range compressor, noting that the dynamic range expander is defined in a similar but complementary way [3]. This leads us to colloquially conflate the two processors, referring to them as simply "the compressor." The compressor is parameterized by its threshold T, ratio r, and time-smoothing con-

stants for attack and release (τ_{att} and τ_{rel}, respectively). First, a logarithmic signal level $\log a_{short}[n]$ is extracted using an analogous running RMS detector as in equation (2) with relatively fast integration time (hence the connection to and the reuse of the $a_{short}[n]$ term). Then, the processor calculates an initial, "raw" gain curve $g_{raw}[n]$ via

$$g_{raw}[n] = \begin{cases} \exp\left[1/r\left(T - \log a_{short}[n]\right)\right] & \text{if } a_{short}[n] > T \\ 1 & \text{otherwise} \end{cases} \tag{4}$$

The gain curve $g_{raw}[n]$ is subjected to time-varying smoothing of the form

$$g[n] = \alpha_{short}[n] g_{raw}[n] + (1 - \alpha_{short}[n]) g[n-1] \tag{5}$$

where

$$\alpha_{short}[n] = \begin{cases} 1 - \exp[-1/(\tau_{att} f_s)], & \text{if } g_{raw}[n] > g[n-1] \\ 1 - \exp[-1/(\tau_{rel} f_s)], & \text{otherwise} \end{cases} \tag{6}$$

yielding the smoothed gain trace $g[n]$. The output signal is computed as $y[n] = c \cdot g[n] \cdot x[n]$, where the additional "make-up" gain parameter c accommodates for the fact that applying compression/expansion to an input signal tends to decrease/increase the overall signal level. The smoothed gain trace can be shifted backwards in time to better anticipate transients (a concept known as *lookahead*). However, we will limit our attention to causal applications throughout the course of this paper.

When its parameters are set correctly, the dynamics range compressor indeed reduces the dynamic range of a signal. However, from a usability perspective, the dynamics range compressor has a number of deficiencies.

1. While the processor as a whole affects the microdynamics of a signal, there is no single parameter that directly affects the microdynamics of the output signal - instead, the parameters interact in a complex way to reduce the microdynamics. For example, lowering the threshold T will have a more dramatic effect on the dynamic range if the ratio parameter r is higher. In this way the compressor fails to provide *direct control* over the microdynamics of the signal.
2. Additionally, the compressor has a *dead-zone*: if the threshold T exceeds the highest level the signal achieves, the processor will have no effect and will always apply unity gain. Changing any parameter other than the threshold in this situation will not have an effect on the output audio.
3. Finally, the processor is *level-dependent* in the sense that applying a static gain to the input signal may cause the processor to have a different effect on the dynamic range. For example, the static gain may cause the input signal to fall entirely beneath the threshold. The processor is additionally sensitive to the macro-dynamics of the input signal. There may not be a single value for the threshold T that will have the desired effect for both the verse and chorus sections of a song if each section has a different average level.

In the next section, we propose a processor for microdynamics that provides direct control, has no dead-zones, and is not affected by level or the underlying macrodynamics of the signal.

3. PROPOSED DYNAMIC RANGE PROCESSOR

Motivated by the LDR, the proposed dynamic range adjusting processor begins by defining a more generalized measure of relative level than in equation (1). This generalization is necessary considering that the resulting level *meter* will now also form the basis for a microdynamics *processor*. The resulting measure effectively allows us to utilize the useful properties of the ratio-based relative level definition in equation (1), while borrowing the notion of separate attack and release ballistics from the compressor in equations (5) and (6). Accordingly, we define relative level for our processor as

$$L[n] = \frac{a_{short}[n]}{a_{long}[n]} = \frac{f_{short}(|x[n]|^p; \tau_\uparrow, \tau_\downarrow)}{f_{long}(|x[n]|^p; s, \tau_\uparrow, \tau_\downarrow)} \tag{7}$$

where p is a constant defining the order of the amplitude envelope (common values include 1 or 2, and we keep to $p = 2$ in this work). The short- and long-term window functions, f_{short} and f_{long}, respectively, are parameterized by user-facing ballistic controls which will be discussed shortly. Ultimately, the proposed processor takes the measured relative levels, as defined by the ballistic controls, and attempts to directly alter them in order to adjust the dynamic range of the input signal. This definition of signal strength will prove critical to ensuring that the behavior of its processing is independent to input level.

The short-term amplitude envelope $a_{short}[n]$ in equation (7) is used to measure more rapid changes in the input signal. Unlike existing measures of microdynamics, the short-term window function f_{short} used here is parameterized by separate time constants for attack and release, $\tau_\uparrow$ and $\tau_\downarrow$, respectively, in a similar manner as in the compressor. It implements the time-varying filtering operation

$$a_{short}[n] = \alpha_{short}[n]|x[n]|^p + (1 - \alpha_{short}[n]) a_{short}[n-1] \tag{8}$$

where

$$\alpha_{short}[n] = \begin{cases} 1 - \exp[-1/(\tau_\uparrow f_s)], & \text{if } |x[n]|^p > a_{short}[n-1] \\ 1 - \exp[-1/(\tau_\downarrow f_s)], & \text{otherwise} \end{cases} \tag{9}$$

The separate attack and release controls allow us to define the rise and fall behaviors of the processor in potentially asymmetrical ways. Figure 2 illustrates the different rise and fall behaviors exhibited when computing $a_{short}[n]$ with different attack and release times on a 125 ms long rectangular pulse.

Meanwhile, the long-term amplitude envelope $a_{long}[n]$ in equation (7) tracks slower changes in the signal envelope. Again, this envelope normalizes the short-term amplitude envelope, which is particularly crucial as we desire our processor to be input-level independent in its behavior. Thus, the relative level metric effectively measures short-term signal energy "in light of" its long-term energy. The ballistics of the long-term window function f_{long} is parameterized by a scale parameter $s > 1$, which ultimately determines a single long-term time constant $\tau_{long} = s \cdot \max(\tau_\uparrow, \tau_\downarrow)$. This conveniently mandates that the long-term window time constant be strictly greater than those used by the short-term window. Accordingly, the long-term window function implements the filtering operation

$$a_{long}[n] = \alpha_{long}|x[n]|^p + (1 - \alpha_{long}) a_{long}[n-1] \tag{10}$$

with smoothing parameter $\alpha_{long} = 1 - \exp[-1/(\tau_{long} f_s)]$.

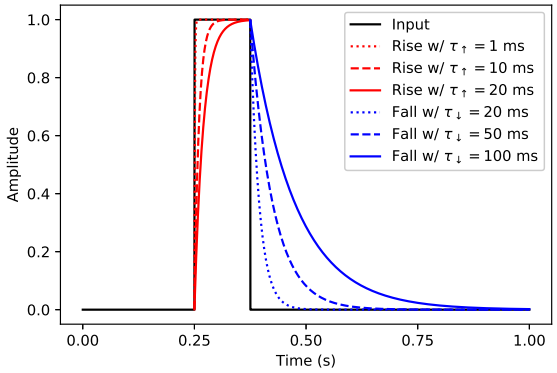


Figure 2: *Short-term amplitude envelopes computed with various attack and release times (with no lookahead).*

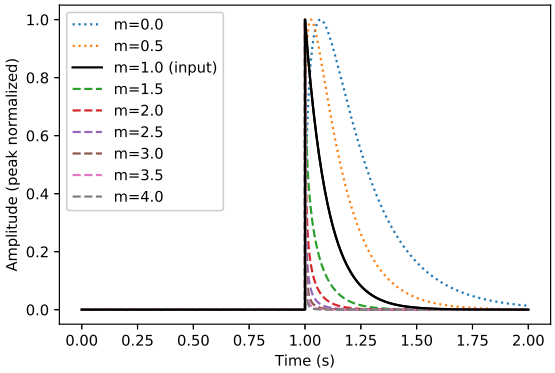


Figure 3: *Processing of an impulse-like signal as a function of m using the proposed processor.*

The processor attempts to directly alter the observed relative levels of the input signal. We consider manipulations of the form

$$\log L'[n] = m \log L[n] \tag{11}$$

where m is a controllable parameter and L' is the altered relative level. This linear change on a decibel scale is similar to a typical compressor or expander transfer curve, except that we are not using any threshold. The normalization feature of our relative levels metric on which we apply this processing to allows us to use such a simple processing model. A gain term is then computed on a per-sample basis, given by

$$\begin{aligned} g[n] &= \left(\frac{L'[n]}{L[n]}\right)^{1/p} \\ &= L[n]^{(m-1)/p} \end{aligned} \tag{12}$$

The output signal is given by $y[n] = c \cdot g[n] \cdot x[n]$, where again, c is a user-facing make-up gain control. Void of any learning-based approach, we have empirically found that c can be "inversely linked" to m to provide some notion of automatic level-matching to the input signal.

With $m > 1$, the processor accentuates microdynamics, working as an expander, whereas with $0 \leq m < 1$, microdynamics are neutralized and the processor works as a compressor. Accordingly, we consider m to function as an "amount" control for the processor, allowing us to dial in the nature of the dynamic range adjustment with a single amount control parameter. Moreover, processing reverts to an identity operation by setting parameters $m = c = 1$. This is illustrated in Figure 3, where we apply the proposed processing on a pulse-like signal with various settings for m. Note that in general, unlike a transient shaper, the processor adjusts the dynamic range of signals without the need to explicitly detect and independently treat transient and sustained signal components. Lastly, the use of the log representation in equation (11) will become more clear when we consider the dynamics matching paradigm in the following section.

The extension of the processor to the multiband case is relatively straightforward, and is motivated by similar extensions of microdynamics measures for multiband analysis and consequent perceptual evaluations [4]. Given a K-band filter bank, the k^{th} crossover extracts a signal sub-band $x_k[n]$ from $x[n]$. The user-facing parametric controls for the processor considered here are the manipulation controls $\{m_k, c_k\}$ and the short-term ballistic controls $\{\tau_{\uparrow,k}, \tau_{\downarrow,k}\}$ at each sub-band k, alongside a single long-term scale control s, leading to sub-band specific long-term window time constants $\tau_{long,k} = s \cdot \max(\tau_{\uparrow,k}, \tau_{\downarrow,k})$. Accordingly, we define relative level measures at each sub-band $L_k[n]$, each with their own ballistics, and altered measures $L'_k[n]$ determined per their independent sub-band manipulation controls. These, in turn, naturally give rise to sub-band specific gain traces $g_k[n]$. The output sub-band is given by $y_k[n] = c_k \cdot g_k[n] \cdot x_k[n]$, and sub-bands are summed to form the output $y[n] = \sum_k y_k[n]$. In a musical context, we can imagine the multiband processor to not only allow for independent processing of potentially different musical elements occupying different ranges of the spectrum, but also to deliver more transparent processing considering that each band is now equipped with its own relative level meter. We refer readers to our demo site at https://sites.google.com/izotope.com/dafx20in22-audio-demo/home.

4. DYNAMICS MATCHING PARADIGM

Given source and target signals $x[n]$ and $z[n]$, respectively, we extend the processor formulation in order to provide a solution to the dynamics matching problem. In this case, we are interested in automatically setting the parameters of the proposed processor so that the microdynamic characteristics of z are transferred to y, the processed version of x. Once again, we begin with the single-band formulation and extend it to the multiband case thereafter. To this end, we assume that the ballistic controls $\tau_\uparrow$, $\tau_\downarrow$, and s are set a priori to some notionally reasonable values (we consider means of relaxing this constraint in the following section). The matching task then involves inference of the m and c parameters.

4.1. Matching criteria

We consider the task of dynamics matching to be one of finding processor parameters whose resulting processed signal $y[n]$ has the same dynamic range as that of the target $z[n]$, as measured by an objective microdynamics measure. As such, we must first define the measure of microdynamics that we will use for our matching criteria. In this work, we opt to use the standard deviation of the logarithm of the relative level $\log L[n]$ as a measure of microdynamics (as indicated by σ_x in Figure 1b). Compared to the LDR, we note that, under a zero-mean Gaussian assumption, order statistics and standard deviations are effectively equivalent to

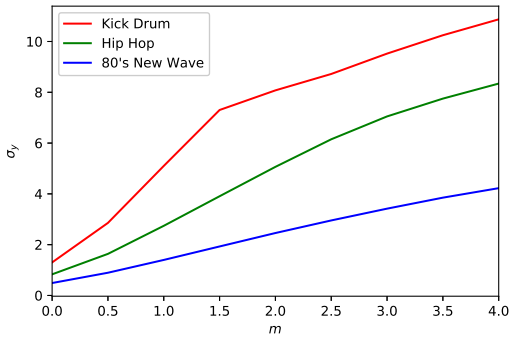


Figure 4: *Output dynamic range σ_y as a function of processing parameter m computed for various musical input signals.*

each other up to a scaling factor. Compared to the DS measure, a key distinction is that we consider not the standard deviations of the short-term amplitude envelope, but rather that of the relative level. We compute standard deviations in an online fashion using Welford's algorithm [19], accommodating learning to take place in either offline or streaming contexts. We informally note that we have also successfully utilized an LDR-like microdynamics measurement (i.e. $\log L_{95}$ or similar maximum statistic) for dynamics matching using this very same algorithm. However, we feel that the use of the standard deviation makes for a clearer presentation of the method, considering Gaussian assumptions for microdynamics distribution that we will make in the following subsection.

4.2. Matching algorithm

We begin by assuming that the logarithms of source and target relative levels are normally distributed with $\log L_x[n] \sim \mathcal{N}(0, \sigma_x^2)$ and $\log L_z[n] \sim \mathcal{N}(0, \sigma_z^2)$, respectively. The zero-mean assumption used here is reasonable when we consider that short-term amplitude envelopes, though momentarily higher or lower than the long-term envelope, tend to deviate around the long-term envelope, and thus, the logarithm of their ratio can be assumed to center around 0. Under the Gaussian assumption, the transformation of random variables described by equation (11) would seemingly result in processed output relative levels $\log L_y[n] \sim \mathcal{N}(0, m^2 \cdot \sigma_x^2)$. Our goal then becomes to match the distribution of $\log L_y[n]$ to that of $\log L_z[n]$, which amounts to matching their standard deviations (conveniently matching our definition of microdynamics). This would appear to be possible by setting $m = \sigma_z / \sigma_x$ (in fact, this was the impetus for developing the proposed processor in the first place). It is evident, however, that when we reanalyze a processed result $y[n]$ that is generated with this setting, this does not exactly hold true in practice. This is to be expected when we consider that both the input signal relative level and the processor output gains are updated at each sample, and that the output gain is not expressed explicitly in terms of the input sample x, but rather in terms of its time-filtered versions. The intuition gained here is still useful, and forms the initial conditions for the dynamics matching routine. Moreover, we note that the resulting output signal microdynamics as a function of m is indeed monotonically increasing, and often appears to be somewhat linear. This is illustrated in Figure 4, where we plot observed standard deviations σ_y as a function of m over a few different input signals.

The behavior illustrated in Figure 4 suggests an efficient search algorithm for m based on an "empirical gradient" technique, as highlighted in Algorithm 1. The resulting method shares some elements to that in [5], with the important distinctions that we are applying the approach to a processor that is better aligned to its (more perceptually relevant) microdynamics measure used as a matching criteria, thus making the nature of our iteration slightly different (and actually more straightforward). We begin by analyzing the microdynamics σ_x and σ_z of $x[n]$ and $z[n]$, respectively. Our initial guess for the amount parameter is effectively $m^{(0)} \leftarrow \sigma_z / \sigma_x$. We denote the processing of $x[n]$ using our proposed processor with pre-determined ballistics and variable manipulation parameters as $y[n] = D(x[n]; m, c)$. If we allow ourselves to generate the processed signal $y^{(0)}[n] = D(x[n]; m^{(0)}, 1)$, and to reanalyze the dynamic range $\sigma_y^{(0)}$ of its result $y^{(0)}[n]$, we can derive an estimate of $\partial\sigma_y / \partial m|_{m=m^{(0)}}$, considering that we already know σ_x and that it corresponds to the "zeroed-state" of our processor with $m = 1$. The resulting slope can be used to suggest a Δm by which we should perturb our current estimate $m^{(0)}$, effectively applying a linear extrapolation considering we desire to produce an output $y[n]$ having microdynamics $\sigma_y = \sigma_z$. We can iterate this scheme until we achieve the desired target microdynamics (within some small tolerance ϵ). Additional constraints can be trivially added in order to limit the number of iterations and/or constrain the range of values that m can take on. Upon inferring a suitable value for m, we measure the RMS of the processed output from the final iteration, and derive a make-up gain amount c such that we match the RMS of x. We process the signal one final time $y[n] = D(x[n]; m, c)$ (with inferred values of m and c), which in this case, is more readily implemented by simply scaling the final iteration's output by the newly inferred value for c. Note that this scaling does not impact the microdynamics measure of the output.

Algorithm 1 Dynamics matching algorithm (single-band case)

$\sigma_y^{(-1)} \leftarrow \sigma_x$
$m^{(-1)} \leftarrow 1$
$i \leftarrow 0$
while $|\sigma_y^{(n-1)} - \sigma_z| \geq \epsilon$ **do**
  if $i = 0$ **then**
    $m^{(i)} \leftarrow \sigma_z / \sigma_y^{(n-1)}$
  else
    $u \leftarrow \left(m^{(i-1)}) - m^{(i-2)})\right) / \left(\sigma_y^{(i-1)} - \sigma_y^{(i-2)}\right)$
    $m^{(i)} \leftarrow m^{(i-1)} + u \cdot \left(\sigma_z - \sigma_y^{(i-1)}\right)$
  end if
  $y^{(i)}[n] \leftarrow D(x[n]; m^{(i)}, 1)$ ▷ Process with current amount
  $\sigma_y^{(i)} \leftarrow \sqrt{Var(\log L_{y^{(i)}})}$ ▷ Analyze processed result
  $i \leftarrow i + 1$
end while
$m \leftarrow m^{(i)}$
$c \leftarrow \sqrt{\sum_n x^2[n] / \sum_n \left(y^{(n)}[n]\right)^2}$
$y[n] \leftarrow D(x[n]; m, c)$ ▷ Dynamics-matched audio

We exemplify the effectiveness of this approach with a simple illustrative example. Specifically, we consider a short musical passage as our input signal $x[n]$, and set the ballistic parameters for our processor/meter to $\tau_\uparrow$ = 1 ms, $\tau_\downarrow$ = 10 ms, and $s = 10$. We create two synthetic target signals z by rendering the input using amounts $m_{opt} = 2.0$ (causing expansion) and $m_{opt} = 0.25$ (causing compression). Note that in general, the algorithm makes no assumptions about the nature of the source and target signals (i.e. the

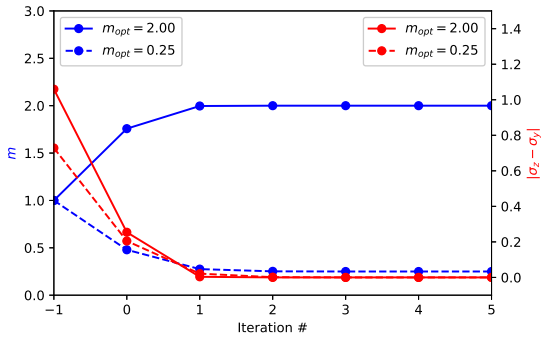


Figure 5: *Illustrative example of the iterative dynamics matching algorithm.*

target signal need not be a rendering of the source signal using our processor). However, this form of target generation gives us access to its underlying ground truth processing parameters. We run our optimization for 5 iterations on each pair of source and target signals, as illustrated in Figure 5. After just a few iterations, we are able to match the microdynamics of both signals, with the inferred amount parameters converging to their respective ground truth values. We observe how the mean absolute difference between processed and target microdynamics approaches zero. Lastly, we note that when matching dissimilar clips to one another, the observed microdynamics will be a function of not only the applied dynamic range processing, but also of performance aspects of the respective recordings. Accordingly, it is possible that these factors get conflated when carrying out the matching algorithm.

Extending the matching algorithm to the multiband case is again straightforward. We found that so long as crossover filters are made steep enough, the core algorithm for determining m_k can be run independently on each sub-band k. We also considered a "simultaneous update" strategy, in which case we consider the vector of sub-band amount controls $\mathbf{m} \in \mathbb{R}^K$ and update its elements jointly. This is effective, but requires the additional computations of combining/filtering into sub-bands at each iteration. Upon determining m_k for each sub-band, we process the signal with the inferred amount controls and unity make-up gain. We compare the RMS values of input and processed sub-bands, and use this to appropriately set c_k. Level-matching each sub-band via the setting of c_k performs a crude version of equalizer matching, which ensures that the resulting multiband processing does not color the signal.

5. DIFFERENTIABLE IMPLEMENTATION

The matching approach outlined in the previous section describes a sound means of determining appropriate processor settings given fixed ballistic conditions known a priori. In order to expand upon this to additionally infer ballistic control parameters, we would need to leverage and/or bootstrap black-box optimization methods (such as the Nelder-Mead [20, 16] or particle swarm optimization algorithms [21, 22]) around the current approach, or construct differentiable versions of dynamic range processors as layers in a deep learning framework, which readily allow gradient flow through them during backpropogation. We consider implementations of the latter, noting the increasingly well-known fact that most any processor which can be expressed as a for-loop of differentiable operations over present inputs and stored states can be constructed exactly in a differentiable framework using recurrent layers [23, 24, 11, 25]. The differentiable extension of such a processor to the multiband case involves several instantiations of the differentiable single-band processor, alongside an implementation of a differentiable crossover network, for which we can use a standard infinite impulse response (IIR) design of Butterworth filters and leverage an efficient implementation for training using the frequency domain sampling method we proposed in [25].

Accordingly, we implement the proposed dynamic range processor, as well as the conventional dynamic range compressor per its definition in Section 2 and [3], as recurrent layers, and use them to create gradient descent optimizers for a matching task. For demonstrative purposes, we simply exemplify the ability to solve the inverse problem of backing out the parameters which were used to process a known input signal. We compare the mean squared error (MSE) between target and inferred waveforms, from which the resulting processing parameters for each respective processor are readily available (and are in fact the only trainable parameters of our "model"). We found that the use of an absolute value activation function was sufficient for enforcing non-negativity of time constants and relevant dynamic range adjusting controls (ratios, amounts, etc.). We use the activation function $s \leftarrow (1+10^{-6})+|s|$ to enforce $\tau_{long,k}$ to be strictly larger than $\tau_{\uparrow,k}$ and $\tau_{\downarrow,k}$ $\forall\ k$.

We run similar experiments for the differentiable realizations of both the proposed processor and the conventional dynamic range compressor, operating on the same input signal $x[n]$. We process $x[n]$ with both processors, using the same numerical values for its attack and release parameters (10 ms and 50 ms, respectively), as well as the same value of 2 for the compressor ratio r and direct amount control m, acknowledging that these parameters result in different output signal behaviors between processors. We do this to acknowledge and verify that the results of optimization are not obscured by considerations around numerical parameter values/ranges between the two processors. Additionally, the standard compressor output was generated using a threshold T of -30 dB, whereas the proposed processor output was generated using $s = 10$, amounting to a long-term integration time τ_{long} of 0.5 seconds. Lastly, make-up gains c are set to unity for both processors, such that no static gain is applied to either signal. The resulting processed outputs serve as the target signals z for each respective processor. We aim to minimize the MSE between the inferred signal y and the target signal z for each processor, whereby y could match z when the underlying processing parameters effectively correspond those used to generate z. Training uses the Adam optimizer for 500 steps with a 10^{-2} learning rate.

The results of our optimizations are illustrated in Figures 6 and 7. The plots in Figures 6a and 7a show that the outputs of the differentiable realizations of the processors match their non-differentiable counterparts. Moreover, the optimization routine, when beginning from random initializations of processing parameters, can be successful for both processors, with optimized solutions matching their intended targets. We also consider running the optimization routines, initializing them from each processor's bypassed state/"zeroed-state." This involves setting compressor threshold T and ratio r and proposed processor amount m to unity. The optimization results for this case are shown in Figures 6b and 7b. While the proposed processor has no problems with this initialization, the compressor optimization fails entirely. This signals that the gradient of the compressor has dead-zones which, as one would imagine, could complicate its use in various optimization tasks. This is verified by Figure 8, which computes the change

in the signal caused by perturbing select parameters from their bypassed state. Indeed, so long as the threshold T is not set below the maximum level of the input $x[n]$ (-20 dB in this case), the output of the compressor does not change, while the proposed processor has a healthy derivative when deviating the value of m from 1.

6. CONCLUSIONS

We proposed a new method for intuitively adjusting the microdynamics of an audio recording. Key to its effectiveness is the inclusion of a relative level measure, whose values we attempt to directly alter during processing. The dynamic range of a signal can be easily increased/decreased primarily via a single amount control. The formulation of the processor is linked to a corresponding microdynamics measure, and we described an iterative dynamics matching algorithm based on this measure. We implemented the processor as a differentiable recurrent layer, and illustrated how it had better implications for gradient flow relative to a similarly constructed compressor. Future work involves developing more efficient differentiable implementations of the processor, noting that the backpropogation of recurrent layers at audio rate can lead to computational bottlenecks. We are also interested in constructing custom objective functions for dynamics matching within a deep learning framework, extending the deep learning-based optimizer to compare microdynamics of dissimilar signals.

7. ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewers for their invaluable comments while preparing this paper.

8. REFERENCES

[1] B. Katz, Ed., *Mastering Audio: The Art and the Science*, Focal Press, Oxford, 2002.

[2] E. Skovenborg, "Measures of microdynamics," *J. Audio Eng. Soc.*, Oct. 2014.

[3] U. Zölzer, Ed., *DAFX: Digital Audio Effects*, Wiley, New York, NY, USA, second edition, 2011.

[4] S. Fenton and H. Lee, "A perceptual model of "punch" based on weighted transient loudness," *J. Audio Eng. Soc.*, vol. 67, no. 6, pp. 429–439, June 2019.

[5] E. Vickers, "Automatic long-term loudness and dynamics matching," *J. Audio Eng. Soc.*, Nov. 2001.

[6] I. Shepherd, E. Grimm, P. Tapper, M. Kahsnitz, and I. Kerr, "Measuring micro-dynamics—a first step: standardizing PSR, the peak to short-term loudness ratio," *J. Audio Eng. Soc.*, Oct. 2017.

[7] W. M. Hartmann, Ed., *Signals, Sound, and Sensation*, Springer, New York, NY, USA, 1998.

[8] C.J. Steinmetz, J. Pons, S. Pascual, and J. Serrà, "Automatic multitrack mixing with a differentiable mixing console of neural audio effects," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2021, pp. 71–75.

[9] M.A. Martínez Ramírez and J.D. Reiss, "Modeling nonlinear audio effects with end-to-end deep neural networks," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2019, pp. 171–175.

[10] A. Sarroff and R. Michaels, "Blind arbitrary reverb matching," in *Proc. Digital Audio Effects (DAFx2020)*, Vienna, Austria, Sept. 2020, pp. 24–30.

[11] S. Nercessian, "Neural parametric equalizer matching using differentiable biquads," in *Proc. Digital Audio Effects (DAFx2020)*, Vienna, Austria, Sept. 2020, pp. 265–272.

[12] S.I. Mimilakis, N.J. Bryan, and P. Smaragdis, "One-shot parametric audio production style transfer with application to frequency equalization," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2020, pp. 256–260.

[13] P. Bhattacharya, P. Nowak, and U. Zölzer, "Optimization of cascaded parametric peak and shelving filter with backpropagation algorithm," in *Proc. Digital Audio Effects (DAFx2020)*, Vienna, Austria, Sept. 2020, pp. 101–108.

[14] D. Sheng and G. Fazekas, "Automatic control of the dynamic range compressor using a regression model and a reference sound," in *Proc. Digital Audio Effects (DAFx2017)*, Edinburgh, Scotland, Sept. 5-9, 2017, pp. 160–167.

[15] D. Giannoulis, M. Massberg, and J.D. Reiss, "Parameter automation in a dynamic range compressor," *J. Audio Eng. Soc.*, vol. 61, no. 10, pp. 716–726, Oct. 2013.

[16] J. Bitzer, D. Schmidt, and U. Simmer, "Parameter estimation of dynamic range compressors: models, procedures and test signals," *J. Audio Eng. Soc.*, May 2006.

[17] M. Comunità, D. Stowell, and J.D. Reiss, "Guitar effects recognition and parameter estimation with convolutional neural networks," *J. Audio Eng. Soc.*, vol. 69, no. 7/8, pp. 594–604, July 2021.

[18] H. Jürgens, R. Hinrichs, and J. Ostermann, "Recognizing guitar effects and their parameter settings," in *Proc. Digital Audio Effects (DAFx2020)*, Vienna, Austria, Sept. 8-10, 2021, pp. 310–316.

[19] B. Welford, "Note on a method for calculating corrected sums of squares and products," *Technometrics*, vol. 4, no. 3, pp. 419–420, 1962.

[20] J.A. Nelder and R. Mead, "A simplex method for function minimization," *J. Comput.*, vol. 7, pp. 308–313, Oct. 1965.

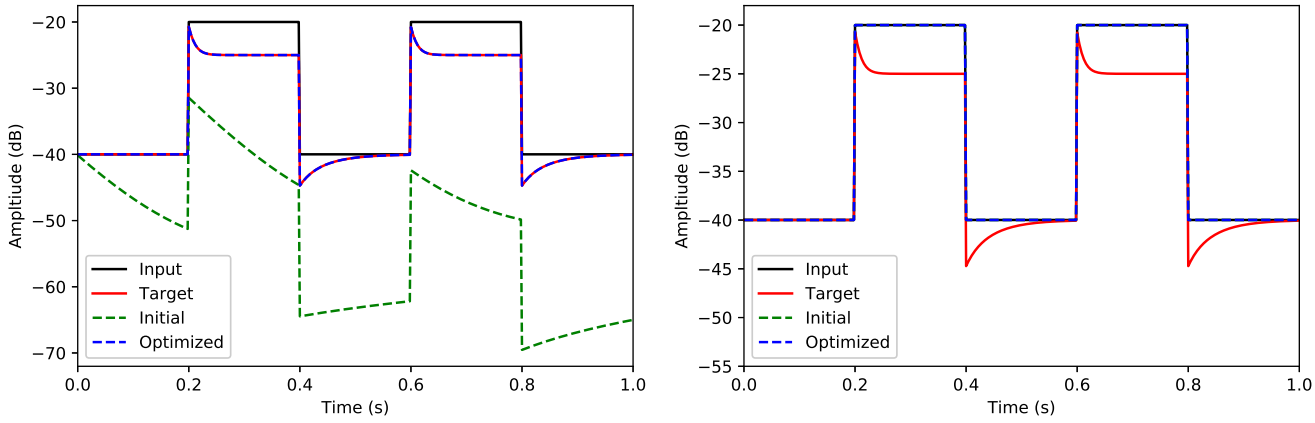
[21] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. of the Int. Conf. on Neural Networks*, 1995, vol. 4, pp. 1942–1948.

[22] D. Moffat, "Objective evaluations of synthesised environmental sounds," in *Proc. Digital Audio Effects (DAFx2018)*, Aviero, Portugal, Sept. 4-8, 2018, pp. 221–228.

[23] F. Esqueda, B. Kuznetsov, and J.D. Parker, "Differentiable white-box virtual analog modeling," in *Proc. Digital Audio Effects (DAFx2020in21)*, Vienna, Austria, Sept. 8-10, 2021, pp. 41–48.

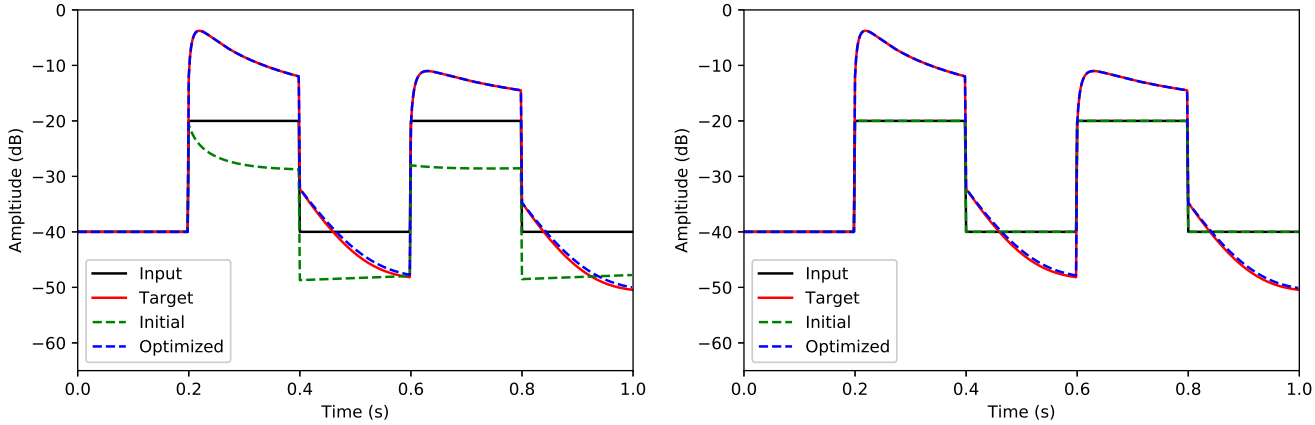
[24] B. Kuznetsov, J.D. Parker, and F. Esqueda, "Differentiable IIR filters for machine learning applications," in *Proc. Digital Audio Effects (DAFx2020)*, Vienna, Austria, Sept. 2020, pp. 297–203.

[25] S. Nercessian, A. Sarroff, and K.J. Werner, "Lightweight and interpretable neural modeling of an audio distortion effect using hyperconditioned differentiable biquads," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2021, pp. 890–894.



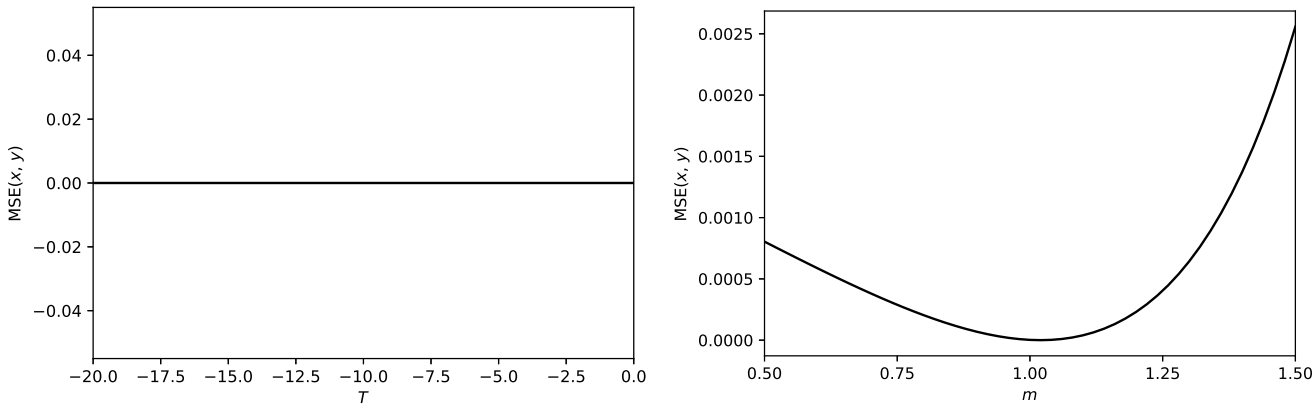
(a) (b)

Figure 6: *Optimization of a differentiable compressor using (a) random initialization and (b) initialization from its "zeroed-state."*



(a) (b)

Figure 7: *Optimization of a differentiable realization of the proposed dynamic range adjusting processor using (a) random initialization and (b) initialization from its "zeroed-state."*



(a) (b)

Figure 8: *Mean squared error between input signal x and its processed signal y, varying the (a) threshold control T of a compressor and (b) amount control m of the proposed processor from their "zero states."*