EVALUATING DYNAMIC RANGE COMPRESSOR MODELS USING CONTROL-VOLTAGE MEASUREMENTS: AN APPROACH AND DATASET

Benjamin R. Thompson and Michael C. Heilemann

Dept. of Electrical & Computer Engineering
University of Rochester
Rochester, USA
bthomp23@ur.rochester.edu

ABSTRACT

The quantity that defines the behavior of a dynamic range compressor is the time-varying gain applied to the signal as a function of the input level. However, models of these devices are typically evaluated using proxy metrics because isolating the gain reduction signal from the audio input–output data included in existing datasets creates an ill-conditioned inverse problem. It is unclear how accurately these metrics describe the behavior the model is tasked with emulating, particularly as waveform-based metrics can be influenced by secondary effects introduced by analog processing and capture, even when those effects are inaudible. We investigate a method of evaluation in which the gain-reduction signal produced by a model is measured directly against a gain-reduction control voltage signal produced by the hardware. To evaluate the efficacy of this metric as a learning objective, a gray-box model is trained using loss computed directly over the gain control signals alongside two models trained using common proxy losses. The models trained using proxy losses did not achieve parity with models trained directly on the gain control signal when evaluated with respect to the underlying control trajectory, and the waveform-domain metrics assigned similar errors to models that were clearly separated by the direct metric. To facilitate further exploration of this method of evaluation, we present a Solid State Logic bus compressor dataset that includes the gain control voltage signal captured alongside the audio output.

1. INTRODUCTION

A dynamic range compressor (DRC) is a processor that applies a time-varying attenuation to a signal as a function of its level. In general, attenuation is only applied when the input level exceeds a threshold set by the operator, and the amount of attenuation is proportional to the amount by which the signal exceeds that threshold. The method used to calculate signal level, the mapping from signal level to attenuation, and the ballistics that govern the application of that attenuation are implementation-specific [1]. As a result, in a creative context, certain DRCs are often preferred for the behaviors that arise from these design details, and it can be desirable to capture the behavior of a specific hardware implementation in a digital model.

The design and analysis of digital DRCs using traditional digital signal processing (DSP) techniques is well established [2, 3, 4]. In recent years, neural approaches to modeling dynamic range compression have been widely explored. These include black-box methods that learn the input–output mapping directly from data [5, 6, 7], as well as gray-box approaches that combine neural networks with structured, differentiable DSP models [8, 9, 10].

To support the training of black- and gray-box models and the validation of white-box models, a number of publicly available datasets for dynamic range compression have been proposed [11, 12, 7], including a recent large-scale collection of data processed by an SSL G-Comp bus compressor [13].

The ideal, discrete version of the relationship that defines a DRC is

$$y_{\mathrm{dB}}[k] = x_{\mathrm{dB}}[k] + g_{\mathrm{dB}}[k], \tag{1}$$

where y_{dB} is the output signal in decibels (dB), x_{dB} is the input signal in dB, g_{dB} is the gain control signal in dB, and k is the sample index.

It follows from Eq. (1) that the accuracy of any model of the ideal system is defined by the accuracy of the quantity $g_{\mathrm{dB}}[k]$. For a hardware ground truth, the signal has been modified by secondary effects, including phase shifts, amplitude-response deviations, and noise contamination, in addition to the intentional amplitude modulation imposed by $g_{\mathrm{dB}}[k]$. As such, extracting $g_{\mathrm{dB}}[k]$ represents an ill-conditioned inverse problem [14] that is further complicated by numerically unstable sample-wise division. As a result, proxy metrics operating over the output waveforms are employed to train and evaluate compressor models. These proxy metrics are sensitive to the same secondary effects that make gain extraction impractical, limiting their ability to reflect error in $g_{\mathrm{dB}}[k]$.

In each of the datasets referenced above, the signals provided are limited to audio input–output pairs where the underlying gain control signal is not directly observable. This work presents a dataset of music and calibration signals processed by an SSL Logic FX G384 bus compressor, in which each audio input–output pair is accompanied by the corresponding gain control signal [15] . The inclusion of $g_{\mathrm{dB}}[k]$ in the dataset allows model performance to be measured directly against the quantity that defines the processor's behavior and further enables proxy metrics themselves to be evaluated against a ground-truth error defined on the gain trajectory.

The SSL bus compressor was chosen both for its relevance in the audio industry, as evidenced by the number of commercial emulations available [16, 17, 18], and because aspects of its implementation are not captured by generic DRC models, including a curved static response and program-dependent release behavior, in which the effective release time is not fixed but instead depends on the recent history of the input signal.

The remainder of this paper is organized as follows. Two proxy metrics commonly used in compressor modeling are discussed in Section 2, and the use of control-signal error to evaluate

the behavioral accuracy of such models is motivated; an experimental comparison of loss regimes is described in Section 3; and the proposed SSL bus compressor dataset and measurement procedure are presented in Section 4.

2. LOSS FUNCTIONS AND EVALUATION METRICS FOR COMPRESSOR MODELS

The principal effect of dynamics processing is a time-varying modification of the signal amplitude. Hardware processors inevitably introduce secondary effects such as frequency-dependent changes in phase and amplitude response and the accumulation of additive noise. When developing a digital model for creative use, these secondary effects can generally be ignored where they are perceptually insignificant or deemed undesirable, or modeled separately where they are considered desirable. However, when evaluating the behavioral accuracy of a DRC model using metrics computed over the processed waveforms, secondary effects and capture artifacts can obscure the true error in the gain-reduction signal.

Consider the discrete outputs for a hardware compressor and a digital model of that compressor, $y[k]$ and $\hat{y}[k]$ respectively. In both cases, the discrete input signal is $x[k]$. A common metric used to quantify the accuracy of a compressor model is L_1, or Mean Absolute Error over the processed waveforms, computed as

$$L_1 = \frac{1}{K}\sum_{k=1}^{K} |y[k] - \hat{y}[k]|\,, \tag{2}$$

for an output signal containing K samples. If we assume that each processor conforms to the ideal model of a compressor, given here as the linear equivalent of Eq. (1):

$$y[k] = g[k]x[k], \tag{3}$$

where $g[k] = 10^{\frac{g_{\mathrm{dB}}[k]}{20}}$, we can rewrite Eq. (2) as

$$L_1 = \frac{1}{K}\sum_{k=1}^{K} |x[k]|\,|g[k] - \hat{g}[k]|. \tag{4}$$

Under these conditions, L_1 represents the average sample-wise error between the gains, weighted by the magnitude of the input signal, making it a useful, if imperfect, proxy. However, its efficacy as a proxy for the error over $g[k]$ is diminished when the ideal compressor model is replaced by one that includes secondary effects:

$$y[k] = g[k](h \circledast x)[k] + N[k], \tag{5}$$

where $\circledast$ is the convolution operator, $N[k]$ is noise, and $h[k]$ is an impulse response that captures the linear effects of the signal chain outside of the broadband gain modulation imposed by the processor. These can include global delay and gain, as well as frequency-dependent group delay and amplitude changes imposed by anti-aliasing filters, buffering, and amplification stages. The L_1 error between an ideal digital processor and a hardware ground truth then becomes

$$L_1 = \frac{1}{K}\sum_{k=1}^{K} |g[k](h \circledast x)[k] + N[k] - \hat{g}[k]x[k]|\,. \tag{6}$$

Under these conditions, the per-sample differences between $(h \circledast x)[k] + N[k]$ and $x[k]$ can dominate the metric even when the effects of the filtering and noise are audibly imperceptible.

Evaluating model performance in terms of L_1 over processed waveforms can be particularly misleading when comparing models that synthesize the processed waveform directly to white- or gray-box signal-processing models, as the former may reproduce secondary effects introduced by the processor or capture chain—such as alterations to phase and/or magnitude response and the addition of noise—that the latter are rarely designed to simulate. This discrepancy is evident in [13], where the smallest L_1 error among the black-box models is nearly an order of magnitude lower than that of the white-box models. When these same models are evaluated using multi-resolution STFT (M-STFT) error [19], which compares short-time spectral magnitudes and is more robust to phase or temporal misalignment, the white-box model outperforms the black-box model. In this case, L_1 error over the output waveform primarily reflects the models' ability to mimic secondary effects of the processor or artifacts of the capture chain, rather than their ability to replicate the gain behavior of the compressor.

The shortcomings of L_1 for quantifying the accuracy of dynamics processor models are further magnified when extracting $g[k]$ explicitly from the hardware input–output pairs. To eliminate division by zero and suppress numerical noise, we define a subset of sample indices that correspond to input signal values above a small threshold, ϵ, given by

$$\mathcal{K}_\epsilon = \{\, k : |x[k]| > \epsilon \,\}. \tag{7}$$

For a digital compressor algorithm, the gain imposed by the model is given by

$$\hat{g}[k] \approx \frac{\hat{y}[k]}{x[k]}, \quad k \in \mathcal{K}_\epsilon. \tag{8}$$

However, applying the same operation to the hardware output in Eq. (5) gives

$$\frac{y[k]}{x[k]} = \frac{g[k](h \circledast x)[k] + N[k]}{x[k]} \neq g[k]. \tag{9}$$

The gain signal is obscured by the noise created by sample-wise division, even under very modest perturbations of the signal outside the intended gain reduction. This limitation is not specific to L_1 over processed waveforms: when quantifying compressor model accuracy, any metric that computes loss per sample over $y[k]$ and $\hat{y}[k]$ is affected by the same sensitivity to secondary effects.

To sidestep the issues inherent with sample-wise waveform loss for this application, Wright and Välimäki [9] propose Multi-Resolution Short-Time Energy (MSTE) loss, in which the processed waveforms are pre-filtered to remove DC and then divided into M overlapping frames of length N_f. The short-time energy (STE) loss is defined as the absolute difference in energy for each frame, averaged across all frames.

$$\epsilon_{STE} = \frac{1}{MN_f}\sum_{m=0}^{M-1}\left|\sum_{n=0}^{N_f-1}\left(y_p[m,n]^2 - \hat{y}_p[m,n]^2\right)\right|, \tag{10}$$

where y_p and $\hat{y}_p$ are the pre-filtered processed waveforms from the hardware and the model, respectively, m indexes the frame, and n indexes the sample with each frame. The MSTE loss is then obtained by evaluating ϵ_{STE} at I values of N_f, and taking the mean across these resolutions:

$$\epsilon_{MSTE} = \frac{1}{I}\sum_{i=0}^{I-1}\epsilon_{STE,i}. \tag{11}$$

While MSTE and related metrics help address the sensitivity of sample-wise metrics to secondary effects such as filtering and the addition of noise, they remain proxies for the underlying quantity of interest. In the context of model evaluation, metrics defined directly on the quantity of interest are preferable, as they represent an interpretable notion of error that is independent of the carrier waveform.

In the context of model training, a proxy metric is useful to the extent that minimizing the metric drives the model toward improved performance with respect to the true objective. A proxy can be functionally equivalent to the desired quantity, in the sense that it yields the same optimal model parameters, only insofar as it preserves the relative ordering of model error across all candidate solutions, for example when the proxy is a monotonic transformation of the true quantity.

Within this framework, MSTE constitutes a useful proxy, as it captures coarse agreement in signal energy across time. However, it is not guaranteed to be functionally equivalent to the underlying control signal for this application. MSTE measures agreement in frame-wise signal energy and may therefore be interpreted as operating on a short-time energy envelope estimate. As such, it is dependent on analysis parameters such as window length and overlap, and reflects a trade-off between temporal resolution and stability. Because envelope estimation from the waveform is not uniquely defined, different estimators or parameter choices may produce different results under identical conditions. Moreover, by aggregating energy within finite windows, MSTE is insensitive to the temporal distribution of energy within each frame, such that distinct gain trajectories can produce similar loss values.

The multi-resolution formulation mitigates these limitations by evaluating the metric at multiple time scales. However, the extent to which temporal discrepancies are resolved depends on the number and density of resolutions considered, and therefore on computational cost. Increasing the number of frame lengths increases the number of computations per training step, which can materially impact time to convergence in large-scale settings. Furthermore, no finite set of resolutions can guarantee equivalence with a model trained using loss computed directly on the quantity of interest.

In this work, we present an SSL bus compressor dataset consisting of input–output signals alongside the corresponding control voltage (CV) signal applied to the voltage-controlled amplifier (VCA). The CV signal included in the dataset is prescaled to correspond exactly with the gain reduction applied by the processor in decibels, $g_{\mathrm{dB}}[k]$. This enables users to process samples in this dataset using an existing model and, provided the model exposes its estimated gain trajectory, evaluate the performance of that model with respect to the principal quantity of the processor. The same measurements can be used to train new machine learning models using loss functions defined on the true objective, eliminating the need for proxy metrics altogether.

An additional advantage of capturing CV directly is that the control signal evolves on much slower time scales than the audio waveform. As a result, aggressive low-pass filtering can be applied without damaging the gain-reduction signal, yielding a measurement with substantially improved signal-to-noise ratio. By contrast, although the gain trajectory is low-frequency, it cannot be isolated from the audio by low-pass filtering without removing the carrier structure required to observe it.

The linear gain signals from a hardware SSL compressor and a digital model are shown in Fig. 1, along with sample-indexed representations of the three error metrics discussed in this paper. For the purposes of visualization, the error generated by each frame in the MSTE calculation was distributed uniformly over the samples in that frame, and the error shown at any sample is the summation of those contributions. The two proxy metrics, absolute error and MSTE over the output waveforms, are significantly noisier than the direct metric, absolute gain error. Additionally, the coarse structures of the proxy metrics track the trajectory of the direct metric only loosely and fail to capture salient structure in the gain-reduction error. This is particularly evident near the three-second mark, where there is a peak in gain-reduction error that is not captured by either of the proxy metrics.

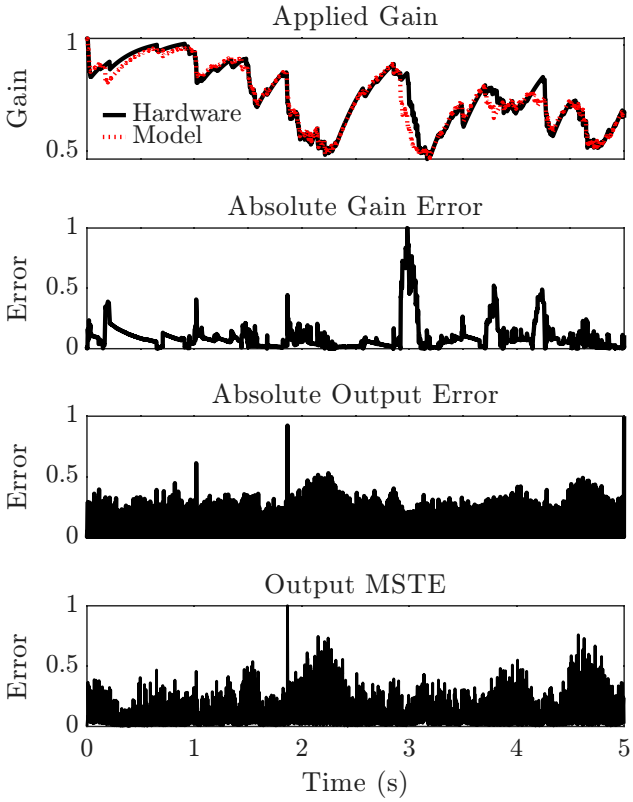

Figure 1: Dynamics processor error metrics. The top panel shows the measured linear gain signal, $g[k]$, from a hardware SSL compressor alongside the gain produced by a software model, $\hat{g}[k]$, for the same input. The second panel shows the absolute error between the linear gain signals, $|g[k] - \hat{g}[k]|$. The third panel shows the absolute error between the output waveforms, $|y[k] - \hat{y}[k]|$. The bottom panel shows the MSTE error between the output waveforms, where each frame's contribution is distributed uniformly over the duration of that frame. The frame sizes used in the MSTE calculation are 8, 16, 32, and 64 samples, with 75% overlap as suggested in [9]. All error signals are normalized by their respective maxima to facilitate comparison.

3. EXPERIMENTAL EVALUATION

To compare each loss regime in context, an existing gray-box dynamics model, `torchcomp` [10], was trained using each of the

losses discussed above: L_1 over gain reduction, MSTE over processed waveforms, and L_1 over processed waveforms. These losses are denoted $L_1(g_{\mathrm{dB}}, \hat{g}_{\mathrm{dB}})$, $\mathrm{MSTE}(y, \hat{y})$, and $L_1(y, \hat{y})$, respectively.

To better align the model with the SSL topology, `torchcomp` was modified by replacing the RMS detector with a peak detector driven by the maximum instantaneous amplitude across channels. This produces a single shared gain control signal, which is applied identically to the left and right input channels. The optimized model parameters were threshold, ratio, attack time, release time, and makeup gain; all other aspects of the `torchcomp` compressor formulation were left unchanged.

It should be noted that there are aspects of the SSL behavior that this model cannot fully reproduce. `torchcomp` employs a static curve with an infinitely hard knee and a constant slope above the nominal threshold. While this is a reasonable approximation of the SSL behavior at the 10:1 ratio setting, its accuracy is diminished at lower ratios, where the transition into gain reduction becomes more gradual and the curvature of the static response above the threshold is more pronounced. Additionally, the ballistics filter in `torchcomp` is a fixed, single-pole design, while the AUTO release setting in the SSL compressor is implemented as a two-pole filter [20], yielding program-dependent release trajectories. For this reason, the training data was restricted to examples measured with the ratio control set to 10:1 and single time constant release settings.

For each loss regime, ten independent models were trained, each using a single 30-second stereo excerpt drawn from the proposed dataset. The selected examples span five hardware control permutations, where each permutation denotes a distinct combination of compressor control settings. For each model, the checkpoint achieving the lowest value of its native training loss was retained. The training examples were selected pseudo-randomly from the eligible examples using a fixed seed, with at most one example drawn from any given song. The same set of ten examples was used across all loss regimes. For the two proxy metrics, prior to loss calculation at each step, the processed waveforms were prefiltered to remove DC using the IIR filter proposed in [9]:

$$H(z) = \frac{1 - z^{-1}}{1 - 0.995 z^{-1}}. \tag{12}$$

Each model was trained independently for 3000 steps using identical optimization settings. The number of steps was chosen empirically to ensure reliable convergence across all loss regimes. The Adam [21] optimizer was employed with a learning rate of 0.01, and a ReduceLROnPlateau scheduler was applied with a factor of 0.5 and a patience of 25 epochs, reducing the learning rate whenever the training loss failed to improve for 25 consecutive updates. All hyperparameters and model configurations were held constant across runs and loss regimes.

The purpose of this experiment is not to evaluate how this particular model generalizes, but rather to compare the extent to which different training objectives recover the underlying gain trajectory under otherwise identical optimization conditions. Accordingly, a traditional train-validation split is not used. Instead, each model is fit to a single measured example and evaluated on that same example in terms of each of the error metrics. This approach isolates the relationship between the optimized loss function and the control-signal objective: if a proxy loss is an effective surrogate for gain-reduction error, minimizing it should recover gain trajectories comparable to those obtained by minimizing gain-reduction error directly.

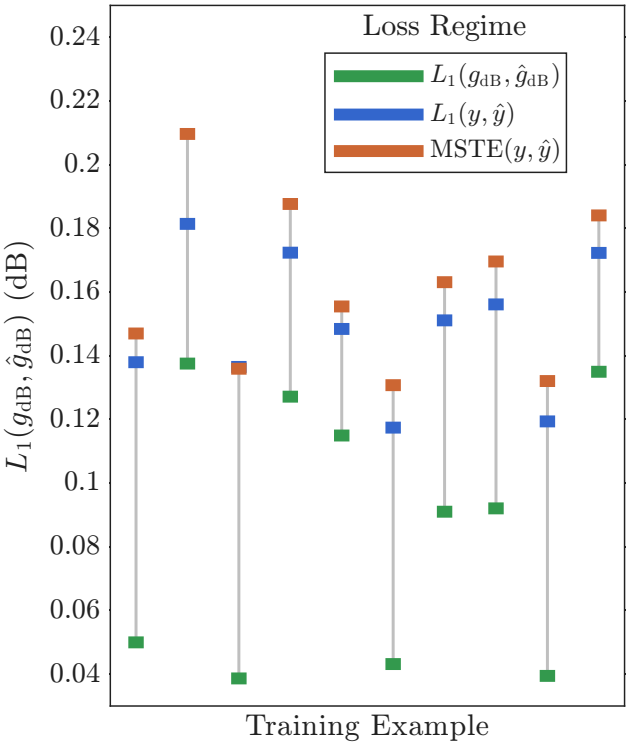

Figure 2: Gain-reduction signal error of trained models for each training example and loss regime. The gray lines connect the error values for the same training example.

Table 1: Training loss regime performance. For each training objective, the table reports mean error across ten models under the direct gain-reduction metric, $L_1(g_{\mathrm{dB}}, \hat{g}_{\mathrm{dB}})$, and two waveform-domain proxy metrics. The waveform-domain metrics are computed after the DC-removal prefilter described above.

	Evaluation Metric		
Training Loss	$L_1(g_{\mathrm{dB}}, \hat{g}_{\mathrm{dB}})$	$L_1(y, \hat{y})$	$\mathrm{MSTE}(y, \hat{y})$
$L_1(g_{\mathrm{dB}}, \hat{g}_{\mathrm{dB}})$	**0.0869** dB	3.328×10^{-3}	2.746×10^{-4}
$L_1(y, \hat{y})$	0.1493 dB	3.237×10^{-3}	2.444×10^{-4}
$\mathrm{MSTE}(y, \hat{y})$	0.1615 dB	3.238×10^{-3}	2.423×10^{-4}

The L_1 gain-reduction error for models trained using each of the three loss regimes and each of the ten training examples is shown in Fig. 2. The mean error across the ten trained models, computed using $L_1(g_{\mathrm{dB}}, \hat{g}_{\mathrm{dB}})$ and both proxy metrics, is presented in Table 1 for each training objective. As expected, each regime yields the lowest error under the metric for which it was optimized. However, the models are more clearly separated by gain-reduction error: the largest gain-reduction error is 85.9% greater than the smallest gain-reduction error, whereas the corresponding error spreads are 2.8% for $L_1(y, \hat{y})$ and 13.3% for $\mathrm{MSTE}(y, \hat{y})$. This suggests that, under these conditions, direct gain-reduction supervision does not merely optimize an idiosyncratic metric at the expense of waveform-domain agreement; rather, evaluating the error through the processed waveform collapses some of the gain-

trajectory structure made observable by the measured control signal, yielding similar waveform-domain errors for models that are clearly distinguishable under the direct metric. With respect to the central question of whether these metrics serve as effective proxies for gain-reduction error, the models trained using waveform-based losses do not achieve parity with the model trained directly on gain reduction when evaluated against the underlying control signal. This indicates that while the proxy losses are correlated with the underlying quantity of interest, minimizing them is not equivalent to minimizing gain-reduction error directly.

4. DATASET

The dataset comprises 219 stereo input clips of 30 seconds each, sampled at 44.1 kHz with 24-bit resolution. For each clip, audio outputs were captured across 90 control permutations, yielding 19,710 stereo output examples. Each output is accompanied by a single-channel control signal sampled at 44.1 kHz with 32-bit resolution. Calibration signals and associated metadata are also included for each control permutation. The total dataset size is approximately 270 GB.

4.1. Hardware

First introduced as part of the center section of the Solid State Logic (SSL) 4000 B console in 1976, the SSL bus compressor has been produced in multiple revisions and form factors [22]. The hardware unit measured for this dataset is a 1991 SSL Logic FX G384 bus compressor [23] shown in Fig. 3. The unit was fully serviced and calibrated prior to measurement. Additional trimmers were added at the control inputs of the audio VCAs to provide the same calibration flexibility available in the console version. The audio path VCAs in this unit are THAT Corporation 202-series devices [24], while the sidechain VCAs are DBX-branded 2151s [25]. As with other bus compressor derivatives, this design primarily differs from other versions of the SSL bus compressor in the input/output circuitry (balancing, debalancing, and line driver implementations) rather than the core dynamics control path [20].

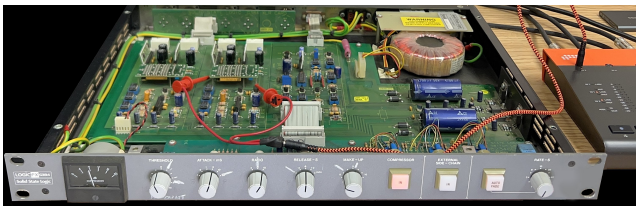

Figure 3: SSL Logic FX G384 bus compressor measured for dataset. The wire test leads are connected to the control input of the left audio VCA.

4.2. Input Signals

4.2.1. Music

As the intended application of this class of compressor is the processing of a full stereo mix [26], the most meaningful input signals are complete songs. Additionally, we require signals that have not already seen heavy dynamic range limiting. To meet these criteria, we use excerpts of the same input signals as in [13], drawn from multitrack recordings accessed via the Cambridge MT search interface [27]. The shared input set provides an opportunity for direct comparison across datasets, isolating differences due to hardware implementation and control settings.

For each source file containing N samples, one 30 s excerpt was selected by drawing the starting sample index k_0 pseudorandomly from the viable range

$$0 \leq k_0 \leq N - 30 f_s, \tag{13}$$

where f_s is the sample rate. Files with other sample rates were resampled to 44.1 kHz, and each excerpt was given 10 ms fades at its boundaries before peak normalization to a maximum absolute sample value of 0.9. The music inputs are comprised of 219, 30 s stereo clips sampled at 44.1 kHz and stored with 24-bit resolution.

4.2.2. Calibration Signals

In addition to the musical input signals, we include several calibration signals processed at each control permutation. These include a slow ramp with an increase in level that is linear in decibels. The ramp consists of a 1 kHz sine-wave carrier, the amplitude of which rises from -48 dB to 0 dB at a rate of $1.6\,\mathrm{dB\,s^{-1}}$. From the compressor's CV response to this input, it is possible to approximate the processor's static curve, provided that the selected attack time is fast relative to the rate of increase of the ramp. Under this condition, the compressor operates in a quasi-static regime, allowing the static input–output relationship to be observed directly. A functional threshold can likewise be identified from the point at which the CV departs from its baseline.

Three burst signals of different durations are also included. As with the ramp, these are constructed from 1 kHz sine-wave carriers and are designed to cross the threshold and drive the compressor into gain reduction. The compressor's CV response to a burst captures its attack and release behavior. In the case of the AUTO release setting, the dependence of the response on burst duration illustrates the program-dependent release mechanism.

The final calibration signal included for each control permutation is a maximum-length sequence with amplitude chosen to remain below the threshold. Because this signal does not engage the gain reduction mechanism, it can be used in conjunction with the corresponding audio output to determine the relative delay between input and output signals, enabling precise time alignment where required.

We also include a global set of calibration signals for use in determining the scaling factor that translates the measured CV signal to gain reduction in decibels. This set is comprised of eleven discrete bursts with amplitudes increasing in one-decibel steps from 4 dB to 14 dB above the threshold, with each burst long enough to allow the ballistics filter to reach steady state so that the static response to that input can be recorded.

To estimate the scaling factor, the responses are trimmed to the static region, gain reduction in decibels is computed from the audio input–output relationship, and a linear model of the form $g_{\mathrm{dB}} = a \cdot \mathrm{CV}$ is fit to the measured data. The slope, a, of this fit defines the conversion factor from CV to gain reduction in decibels.

Note that this factor has already been applied to the measured CV signals in the dataset, so the provided CV signals are expressed directly as gain reduction in decibels. The calibration signals are included for completeness and to allow end users to reproduce the scaling factor calculation if desired.

Table 2: Control parameters and available settings.

Control	Resolution	Unit	Settings
Threshold	Continuous	dB	$[-15, 15]$
Attack	Switch	ms	0.1, 0.3, 1, 3, 10, 30
Ratio	Switch	dB/dB	2, 4, 10
Release	Switch	s	0.1, 0.3, 0.6, 1.2, Auto
Makeup Gain	Continuous	dB	$[-5, 15]$

4.3. Control Permutations

The relevant user-facing controls on the Logic FX G384 compressor, along with the range of settings for each, are shown in Table 2. To balance coverage with dataset size, all permutations of the switched controls were included for a single threshold and makeup gain setting. Multiple settings for these parameters were not included, as neither represents an independent degree of freedom.

In the hardware unit, makeup gain is implemented by applying a constant DC offset to the control signal that determines the gain of the audio VCA, while leaving the gain of the sidechain VCA unchanged. This results in a simple scaling of the output waveform. As the control signal is supplied in dB, additional permutations can be synthesized by applying the desired change in makeup gain as an offset in dB to both the control signal and the output audio.

Similarly, the threshold parameter enters the gain law only as an additive offset in the dB-domain sidechain VCA control signal. Applying a constant offset Δ to the input signal in decibels,

$$x_{\mathrm{dB}}[k] \mapsto x_{\mathrm{dB}}[k] + \Delta, \tag{14}$$

is equivalent to replacing the threshold parameter with an effective threshold,

$$T_{\mathrm{eff}} = T - \Delta, \tag{15}$$

leaving the control trajectory $g_{\mathrm{dB}}[k]$ unchanged. To preserve consistency of the input–output pair, the same offset must also be applied to the audio output signal,

$$y_{\mathrm{dB}}[k] \mapsto y_{\mathrm{dB}}[k] + \Delta. \tag{16}$$

Accordingly, additional threshold conditions can be synthesized by applying a common gain offset to both the input and output audio signals while leaving the control signal unchanged.

4.4. Captured Data

The responses to each input were captured using a Bitwig Connect 4/12 audio interface [28] at a sample rate of $44.1\,\mathrm{kHz}$. This interface includes the DC-coupled inputs necessary to capture CV signals.

Stereo audio outputs were stored at a resolution of 24 bits per sample. The raw captured audio output reflects any gain applied by the gain cell in the processor along with any additional gain in the signal chain. To simplify interpretation, any static input–output gain that is not explained by the CV signal was removed prior to storage. The factor by which it was adjusted is stored in the metadata, allowing recovery of the original signal if required.

The stored CV is scaled so that it directly represents gain reduction in dB, as outlined in subsection 4.2.2. Since the signal controlling both audio VCAs is identical up to a small calibration scalar, the CV from a single VCA was captured. The CV signals were stored at a resolution of 32 bits per sample to accommodate the large amplitudes expected. Note that while the CV signals were saved in the WAV format, they are not audio signals; they can contain large DC components and are not restricted to the range $[-1, 1]$. Appropriate caution should be exercised such that these signals are not routed to amplifiers or loudspeakers.

4.5. Temporal Alignment of Measured Signals

At capture, care was taken to measure and compensate for system latency. As a result, the input and output audio signals are aligned in time to within one sample.

All signals within a given control permutation were captured using a single instance of a MATLAB `audiostreamer` object, so any residual capture offset is consistent across the recorded channels for that permutation [29]. Sub-threshold MLS calibration signals are provided for each permutation to enable users to verify or refine audio signal alignment if desired.

A procedure is provided here for confirming or refining the alignment of the control signal to the audio signals, should that be necessary for a user's application. The measured CV is converted to a linear gain, and the integer lag that minimizes the discrepancy between the measured output and the gain-scaled input is selected. Specifically, the estimated lag ℓ^* is given by

$$\ell^* = \underset{-L \le \ell \le L}{\arg\min} \frac{1}{|\mathcal{K}_\ell|} \sum_{k \in \mathcal{K}_\ell} \left| y(k) - g(k+\ell)x(k) \right|, \tag{17}$$

where L defines the maximum lag considered in samples, and $\mathcal{K}_\ell$ denotes the set of indices for which the shifted signals overlap. After estimating ℓ^*, the control signal can be shifted accordingly. This approach can be extended to estimate fractional-sample offsets by allowing non-integer lags and evaluating the objective under interpolation.

4.6. Metadata

In addition to the control settings described in Table 2, several derived quantities are included in the dataset metadata to facilitate interpretation and reuse.

In the SSL topology, the effective threshold of the processor is determined by a combination of the offset voltage applied to the sidechain VCA through the threshold potentiometer and an offset current applied at the slope amplification stage. Since the value of the current offset is tied to the position of the ratio switch, the input level at which compression begins is not determined solely by the threshold control.

A derived effective threshold is provided in units of dB, defined operationally from the compressor response to the ramp calibration signal described in Section 4.2.2. The control signal is analyzed in a region below the threshold to estimate its baseline and variance, and the threshold is taken as the input level at which the control signal first exceeds this baseline by five standard deviations. This response-defined threshold corresponds to the onset of observable gain reduction in the measured unit. Because the onset of compression is gradual, particularly at lower ratios, this quantity should not be interpreted as equivalent to the threshold parameter of an idealized static-curve model. Users interested in parameter recovery may therefore wish to transform or redefine the threshold according to the parameterization of the model under study.

For reproducibility and completeness, the nominal threshold setting is also included in the metadata. Because the front-panel

markings associated with the threshold control are sparse, this quantity is reported as the measured voltage at the wiper of the potentiometer.

For the same reason, the makeup gain setting is likewise reported as the measured voltage at the wiper of its corresponding potentiometer. A second makeup gain quantity is also included in the metadata that is derived from the CV signal when the processor is not actively in gain reduction. It is this second value that corresponds to the actual static gain applied by the processor in dB.

In addition to the nominal attack time and release time settings, the corresponding component values in the ballistics filter are included along with the associated time constants. These quantities may be used in the context of a circuit-informed model to predict the expected ballistics behavior and to verify model outputs in a parameter recovery experiment. It should be noted, however, that the effective attack times are greatly accelerated by the loop gain of the control circuit. Additionally, because loop gain is not constant, the attack trajectory deviates from an ideal exponential response.

5. CONCLUSIONS

In this work, we identified the gain control signal, $g_{\mathrm{dB}}[k]$, as the primary quantity of interest when evaluating the accuracy of a DRC model's gain-reduction behavior. In existing DRC datasets, this quantity is not directly observable from the included input–output signals, and models are trained using proxy metrics computed over the processed waveforms. We characterized the limitations of common proxy loss regimes, including L_1 and MSTE computed over the processed audio, which can be influenced by secondary effects such as phase shifts and noise, even when these effects are perceptually insignificant. To evaluate these metrics in context, we trained a differentiable compressor model under each loss regime. When evaluated with respect to the ground-truth gain trajectory, models trained using the proxy losses did not achieve parity with the model trained by minimizing error in $g_{\mathrm{dB}}[k]$ directly, indicating that they are not equivalent surrogates for gain-reduction error in this setting. Additionally, the waveform-domain metrics assigned comparatively similar errors to models that were clearly separated by the direct gain-reduction metric, suggesting that these proxy objectives can obscure some of the salient processor behavior.

We also introduced a dataset of measured music and calibration signals processed by a hardware SSL bus compressor that includes the corresponding gain-reduction signal, $g_{\mathrm{dB}}[k]$, in the form of CV measured alongside the audio signal, enabling direct supervision of the gain reduction and allowing proxy metrics to be evaluated against a ground-truth measure of error. Future work will extend this approach to additional hardware processors and explore models that capture behavior specific to the SSL compressor that generic models cannot replicate.

6. ACKNOWLEDGMENTS

The authors thank Ric Ocasek for bequeathing the hardware used in this work to the University of Rochester Audio and Music Engineering program. This work was supported by a Research Initiation Award from the SoundSpace Institute.

7. REFERENCES

[1] D. Self, *Audio Engineering Explained*, 1st ed. Focal Press, 2010.

[2] D. Giannoulis, M. Massberg, and J. D. Reiss, "Digital dynamic range compressor design—a tutorial and analysis," *Journal of the Audio Engineering Society*, vol. 60, no. 6, pp. 399–408, 2012.

[3] P. Dutilleux, K. Dempwolf, M. Holters, and U. Zölzer, "Nonlinear processing," in *DAFX: Digital Audio Effects*, 2nd ed., U. Zölzer, Ed. Chichester, UK: John Wiley & Sons, 2011, ch. 4.

[4] J. S. Abel and D. P. Berners, "On peak-detecting and rms feedback and feedforward compressors," in *Audio Engineering Society Convention 115*. Audio Engineering Society, 2003.

[5] S. H. Hawley, B. Colburn, and S. I. Mimilakis, "Signaltrain: Profiling audio compressors with deep neural networks," in *Proceedings of the 147th Audio Engineering Society Convention*. New York, NY, USA: Audio Engineering Society, 2019.

[6] C. J. Steinmetz and J. D. Reiss, "Efficient neural networks for real-time modeling of analog dynamic range compression," in *Audio Engineering Society Convention 152*. Audio Engineering Society, 2022.

[7] R. Simionato and S. Fasciani, "Fully conditioned and low latency black-box model of analog compression," in *Proceedings of the International Conference on Digital Audio Effects (DAFx23)*, Copenhagen, Denmark, 2023.

[8] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "Ddsp: Differentiable digital signal processing," in *International Conference on Learning Representations*, 2020.

[9] A. Wright and V. Välimäki, "Grey-box modelling of dynamic range compression," in *Proceedings of the 25th International Conference on Digital Audio Effects (DAFx20in22)*, 2022, pp. 304–311.

[10] C.-Y. Yu, C. Mitcheltree, A. Carson, S. Bilbao, J. D. Reiss, and G. Fazekas, "Differentiable all-pole filters for time-varying audio systems," in *Proc. 27th Int. Conf. Digital Audio Effects (DAFx24)*, Guildford, UK, 2024, pp. 345–352. [Online]. Available: https://dafx.de/paper-archive/2024/papers/DAFx24_paper_75.pdf

[11] M. Comunità, C. J. Steinmetz, and J. D. Reiss, "Differentiable black-box and gray-box modeling of nonlinear audio effects," p. 1580395, 2025. [Online]. Available: https://doi.org/10.3389/frsip.2025.1580395

[12] M. Comunità, C. J. Steinmetz, H. Phan, and J. D. Reiss, "Modelling black-box audio effects with time-varying feature modulation," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[13] Y. Gu, R. Zhang, L. Juvela, and Z. Wu, "Solid state buscomp: A large-scale and diverse dataset for dynamic range compressor virtual analog modeling," in *Proc. 28th Int. Conf. Digital Audio Effects (DAFx25)*, Ancona, Italy, 2025.

[14] A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill-Posed Problems*. Washington, DC: V. H. Winston & Sons, 1977.

[15] B. R. Thompson and M. C. Heilemann, "SSL bus compressor control voltage dataset," Figshare, 2026, dataset. [Online]. Available: https://doi.org/10.60593/ur.d.31892026

[16] Waves Audio, "SSL g-master buss compressor," https://www.waves.com/plugins/ssl-g-master-buss-compressor, n.d., accessed: 2026-03-28.

[17] Universal Audio, "SSL 4000 g bus compressor plug-in," https://www.uaudio.com/products/ssl-4000-g-series-bus-compressor, n.d., online; accessed 2026-03-28.

[18] Solid State Logic, "SSL native bus compressor 2," https://store.solidstatelogic.com/plug-ins/ssl-native-bus-compressor-2, n.d., online; accessed 2026-03-28.

[19] X. Zhang, L. Xue, Y. Gu, Y. Wang, J. Li, H. He, C. Wang, S. Liu, X. Chen, J. Zhang *et al.*, "Amphion: an open-source audio, music, and speech generation toolkit," in *2024 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2024, pp. 879–884.

[20] Solid State Logic, *SSL FX G384 Compressor Schematics*, 1991, main card and switch card schematics, drawing numbers T82345.71–T82345.73 and T82346.71.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[22] Solid State Logic, *The Bus Compressor User Guide*, 2022. [Online]. Available: https://www.solidstatelogic.com/assets/uploads/downloads/The_Bus_User_Guide_1.1.pdf

[23] ——, *Logic FX G384*, Solid State Logic, 1990, user manual.

[24] THAT Corporation, *THAT 202 Series Voltage-Controlled Amplifiers Datasheet*, THAT Corporation, 2000. [Online]. Available: https://thatcorp.com/datashts/202data.pdf

[25] ——, *THAT 2150 Series Voltage-Controlled Amplifiers Datasheet*, THAT Corporation, 1996. [Online]. Available: https://thatcorp.com/datashts/THAT_2150-Series_Datasheet.pdf

[26] Solid State Logic, *SSL G-Series Bus Compressor Module for 500 Series Racks User Guide*, Solid State Logic, 2021. [Online]. Available: https://www.solidstatelogic.com/assets/uploads/downloads/SSL_500_Series_G_Comp_Module_User_Guide.pdf

[27] University of Cambridge, "Cambridge mt multitrack search interface," 2015. [Online]. Available: https://multitracksearch.cambridge-mt.com/ms-mtk-search-ads.htm

[28] Bitwig GmbH, *Bitwig Connect 4/12 User Guide*, Bitwig GmbH, 2025. [Online]. Available: https://downloads.bitwig.com/documentation/connect/Bitwig%20Connect%204-12%20User%20Guide.pdf

[29] MathWorks, "audiostreamer," MATLAB Audio Toolbox Documentation, 2025, accessed: 2026-03-26. [Online]. Available: https://www.mathworks.com/help/audio/ref/audiostreamer.html