# NEURAL MODELING OF MAGNETIC TAPE RECORDERS

*Otto Mikkonen, Alec Wright, Eloi Moliner and Vesa Välimäki\**

Acoustics Lab, Department of Information and Communications Engineering
Aalto University, Espoo, Finland
`firstname.lastname@aalto.fi`

## ABSTRACT

The sound of magnetic recording media, such as open-reel and cassette tape recorders, is still sought after by today's sound practitioners due to the imperfections embedded in the physics of the magnetic recording process. This paper proposes a method for digitally emulating this character using neural networks. The signal chain of the proposed system consists of three main components: the hysteretic nonlinearity and filtering jointly produced by the magnetic recording process as well as the record and playback amplifiers, the fluctuating delay originating from the tape transport, and the combined additive noise component from various electromagnetic origins. In our approach, the hysteretic nonlinear block is modeled using a recurrent neural network, while the delay trajectories and the noise component are generated using separate diffusion models, which employ U-net deep convolutional neural networks. According to the conducted objective evaluation, the proposed architecture faithfully captures the character of the magnetic tape recorder. The results of this study can be used to construct virtual replicas of vintage sound recording devices with applications in music production and audio antiquing tasks.

## 1. INTRODUCTION

Magnetic recording has had a profound impact on the history of recorded music, providing a dramatic leap in the quality of the stored audio in comparison to the earlier direct-to-disk techniques. The advances in magnetic recording grounded practices such as multitrack and sound-on-sound recording within the industry, and as the technology matured and became cheaper, allowed for entire generations of professional and amateur musicians alike to experiment with these powerful production techniques. Reel-to-reel tape recorders, an example of which is shown in Fig. 1, have been largely replaced by digital recording techniques for sound capture and reproduction. However, the idiosyncrasies of the magnetic recording process are now used as a creative effect. This paper studies the imperfections of the magnetic recording process and emulates them digitally and with neural networks.

Virtual analog (VA) modeling is an area of digital signal processing with a rich lineage in the past decades, aiming at modeling analog audio devices and making the emulations available as software [1]. The techniques used for the modeling are traditionally divided into white-box, grey-box, and black-box methods, depending on the type of information used as the basis for the task.

Figure 1: *Akai 4000D reel-to-reel tape recorder.*

White-box techniques use exact knowledge of the underlying circuits to reconstruct the physics in the digital domain in order to match the observed behavior. Black-box techniques use observations collected from the target as the basis and optimize a general-purpose method to replicate the behavior. In grey-box modeling, a combination of white- and black-box methods is used. Over the last decade, advances in deep learning have given rise to their increased application also for VA modeling, with the techniques capable of exhibiting state-of-the-art performance in a number of different tasks [2, 3, 4].

While the physics underlying magnetic recording has been studied thoroughly in the past [5, 6], it has been applied for the purpose of digitally emulating the recording process only recently [7]. Pertinent to this topic is the lineage of work related to the modeling of tape delay devices, which has been covered more extensively [8, 1, 9]. In our work, we build up from earlier literature on emulating the sound of the magnetic tape, and propose a grey-box system for the emulation task. We consider the sound of the tape recorder to be build up of the nonlinear hysteretic magnetization of the tape medium, the filtering produced by the recording and playback heads, a fluctuating delay induced by the imperfections in the tape transport mechanism, the tape hiss, as well as the subtle nonlinearities and filtering of the input/output amplifiers. The proposed system uses a hybrid array of modern deep-learning techniques as the backbone for modeling these different aspects of the character.

The rest of this paper is organized as follows. Sec. 2 reviews the background theory regarding the effects of magnetic recording. Sec. 3 gives an overview of the system proposed for the modeling. Sec. 4 provides details concerning the data used to evaluate the proposed method, while Sec. 5 describes the training process for the different network architectures. The experimental procedure is divided into two sections depending on the type of data used for the evaluation: in Sec. 6, toy data collected from a virtual tape machine is used, while Sec. 7 uses data from a real machine. Sec. 8 concludes the work.
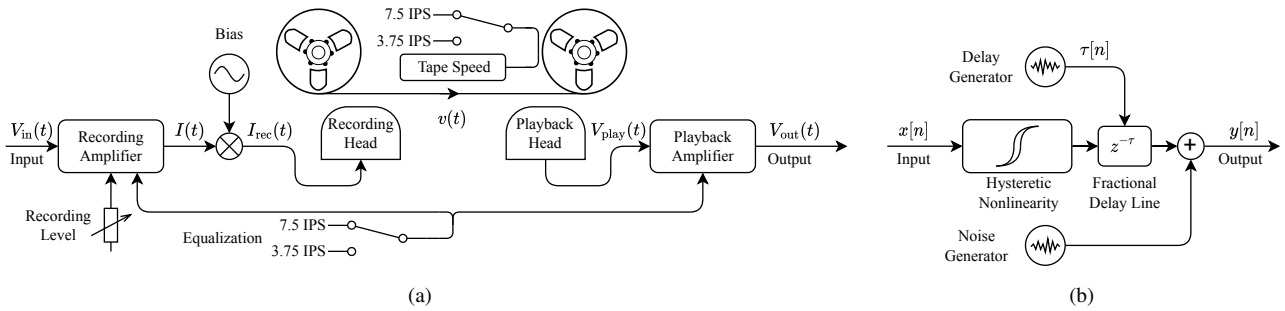
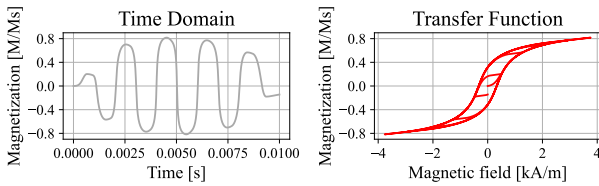Figure 2: *(a) Real and (b) VA system block diagrams.*



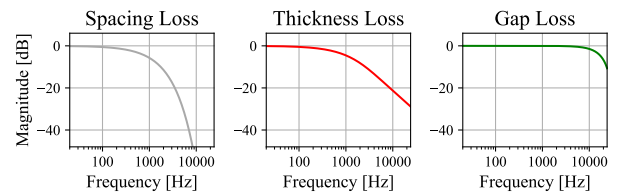Figure 3: *Tape magnetization nonlinearity exhibits hysteresis.*



Figure 4: *Three components of playback losses.*

## 2. MAGNETIC RECORDING

The block diagram of a typical magnetic recorder is shown in Fig. 2a. The system consists of a recording amplifier, a recording head, the moving tape medium, a playback head, and a playback amplifier. The following paragraphs briefly discuss each of these components and their contribution to the overall character, based on earlier literature [7, 5, 6, 1, 8, 9].

The recording head takes the input current from the recording amplifier, and produces a spatial magnetic field determined by both the properties of the recording head and the magnitude of the input current. When the moving magnetic tape is exposed to this field, the magnetic dipoles in the substrate take the form of the field, which is retained as it passes the volume of the field. Since the characteristic magnetization of the tape takes a hysteretic nonlinear shape, shown in Fig. 3, a high-frequency bias is added to the input signal to reduce the nonlinearity produced by the hysteresis.

As the tape moves past the playback head, the changing magnetic field induces a current within its internal coil, restoring part of the stored signal into electrical form. This recovery of the signal is a spatial integration over the magnetized volume of tape, which leads to tape-speed-dependent filtering, shown in Fig. 4. The filtering consists of components related to the spacing between the head and the tape, the tape thickness, and the playback head gap.

To counterweight the spatial filtering induced by the playback head, the recording and playback amplifiers are used as pre- and post-filtering stages. This also serves to maximize the dynamic range of the tape and condition the input and output signals of the system. In our reference design shown in Fig. 1, these amplifiers are implemented with cascaded transistor circuits, producing additional low-order harmonic distortion in the processed signal.

The tape movement speed is not perfectly constant, due to small fluctuations produced by imperfections in the tape transport mechanics. These imperfections include cyclical components produced by the moving parts in the transport mechanism as well as stochastic behavior, shown in Fig. 5a, in the form of a delay trajectory. These inconsistencies in the movement can be heard as small fluctuations in pitch, known as *wow* and *flutter*.

Magnetic tape recorders generate noise artifacts throughout the recording and playback stages, which contribute to the distinctive qualities of the resulting audio signal. The noises could be originated by multiple sources, such as the playback equipment, the magnetic particles in the tape coating, modulation noises during recording, or surface asperities, among others [10].

## 3. METHODS

We employ a grey-box model, shown in Fig. 2b, inspired by the block diagram of the target device. The signal path for the proposed system consists of three components: 1) a hysteretic nonlinearity for modeling the magnetic recording process lumped with the record/playback amplifier responses, 2) a time-varying delay line controlled by a delay trajectory generator, and 3) an additive noise component. In the following, details concerning these components as well as the capture of delay trajectories are given.

### 3.1. Delay Trajectory Retrieval

The fluctuating time delay between the record and playback heads is captured using a pulse train-based measuring technique [8, 11, 7]. The measurement signal consists of a train of unit impulses spaced $T = 1/f$ apart, where $f$ is the repetition frequency. This signal is played through the target device, and the locations of the input and output pulses are compared to determine the time delay between the heads as it fluctuates over time. The frequency $f$ of the pulse train determines the sampling rate for the captured trajectories and in our experiments $f = 100$ Hz was used [8, 9]. An example of a measured delay trajectory, as well as the measurement signal at the input and output of a studied target are illustrated in Fig. 5.

### 3.2. Lumped Nonlinearities

The lumped effects include the hysteretic nonlinearity of the magnetization process, the spatial filtering of the playback head, as
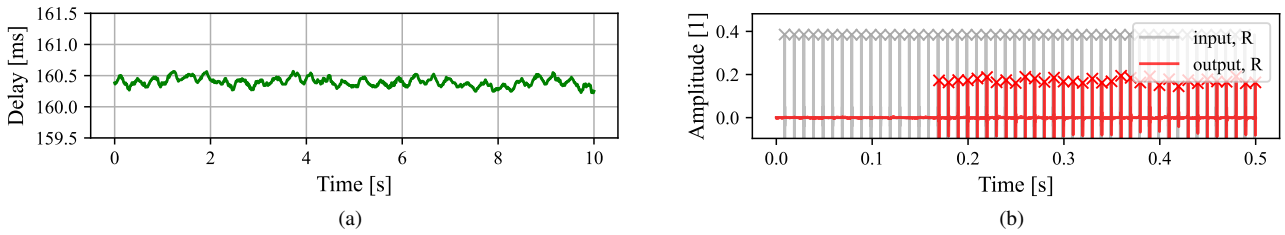
Figure 5: *(a) Delay trajectories and (b) measurement signal.*

well as the low-order distortion components and filtering originating from the record and playback amplifiers. We choose a recursive neural network (RNN) architecture for modeling these aspects, consisting of a gated recurrent unit and a linear output layer [2]. The model details can be found in the original research article. The choice of the model stems from the stateful nature of the recurrent unit, which we hypothesize as being helpful in learning the hysteresis shape. Three training schemes for training the RNN are considered: two supervised and one adversarial, explained in the following subsections.

### 3.2.1. Supervised Approaches

For the supervised approaches, we exploit the stereo nature of most tape recorders. We construct stereo training signals with the audio content on the left channel embedded with the measurement pulse train signal on the right. Using the pulse detection algorithm to construct the delay trajectories for each individual audio segment, the time evolution of the tape medium can be captured and used to restore the alignment of the targets and model predictions.

We implement and compare two approaches for restoring the alignment, shown in Fig. 6. In the first approach (Fig. 6a), inspired by Kaloinen's work [9], we use the captured delay trajectory $\boldsymbol{\tau}$ to demodulate the target segment $\mathbf{y}_L^* = \text{demod}(\mathbf{y}_L, \boldsymbol{\tau})$, where $^*$ denotes a signal which has been demodulated or has not had a delay line applied to it. After this procedure, the demodulated signal and the raw output from the nonlinear block become aligned and we compute the loss $\mathcal{L}(\hat{\mathbf{y}}_L^*, \mathbf{y}_L^*)$. In this scenario, the gradients flow only through the nonlinear block, and the time-varying delay line is only added during inference.

In the second approach, shown in Fig. 6b, we use the captured delay trajectory $\boldsymbol{\tau}$ to delay the raw output from the nonlinear block $\hat{\mathbf{y}}_L[n] = \hat{\mathbf{y}}_L^*[n - \boldsymbol{\tau}[n]]$ in order to compute the loss directly as $\mathcal{L}(\hat{\mathbf{y}}_L, \mathbf{y}_L)$. In this approach, the gradients flow also through the delay line, which needs to be differentiable. Linear interpolation was used to implement the continuously variable delay line, and the implementation was originally included in the Magenta DDSP codebase [12]. Care must be taken to avoid non-differentiable rounding operations such as the floor operator.

The delay line is initially filled from left to right with integers representing delay-line length, starting from $N$ and decreasing with a step size of 1, with the final value being 0. The desired delay-line length is subtracted from each element of this vector and the abs operator is applied. Each element of this vector is subtracted from 1, and finally, the rectified linear unit (ReLU) function is applied, resulting in a vector where all indices are filled with zeros, except for the indices with indexes immediately above and below the desired delay-line length. This vector can then be multiplied element-wise with a buffer of previous input values, and the sum of this operation produces the output of the delay line.

### 3.2.2. Adversarial Approach

An alternative approach using an adversarial training method is also proposed. This allows for the modeling of monophonic tape recorders. In this method, measured or synthesized delay lines are applied to the RNN model output. Instead of training the model using a supervised loss function, which requires the delay line to be known, an adversarial loss can be used such that the delay line applied does not have to match the actual delay line applied in the training data. In this case, a discriminator model is trained to distinguish between real examples of processed audio from the target dataset, and synthetic examples which are produced using our modeling approach. The discriminator model and training procedure used are identical to those used earlier by Wright et al. [13]. The discriminator receives a time-frequency representation of the audio as input, and consists of a stack of 1D convolutional layers, with the first layer treating the frequency bins as an input channel. The discriminator and the tape model are trained adversarially using the hinge loss function.

### 3.3. Noise Generator

The background noise component is modeled using a diffusion probabilistic model [14], in a similar fashion to previous work from Moliner and Välimäki [3]. A diffusion model is used as a data-driven universal approximator of the probability distribution of all the additive disturbances that are introduced during the recording, magnetization, and playback processes. The model can be trained with recorded silent passages containing only the background textures produced by the reel-to-reel machine. By reversing a diffusion process, white Gaussian noise segments are progressively morphed into noises from the training data distribution. Based on the assumption that the noises are additive, the generated noise samples are added to the output signal as a final step.

Although we have adopted the main concept from a previous work [3] as a basis for our research, there are significant deviations in the technical details of our approach due to our use of more recent developments on diffusion models. We adopt some of the design choices from Karras et al. [15], including the ordinary differential equation (ODE) formulation, the neural network preconditioning, the training objective, and the noise schedule parameterization, the latter being a Variance Exploding noise schedule [16].

### 3.4. Trajectory Generator

Given the stochastic nature of the delay trajectories underlined in Sec. 2, in this work, they are modeled using a probabilistic generative model. Similar to Sec. 3.3, a diffusion model is also used to for this task. In this case, the model is trained to emulate the distribution of the measured delay trajectories.
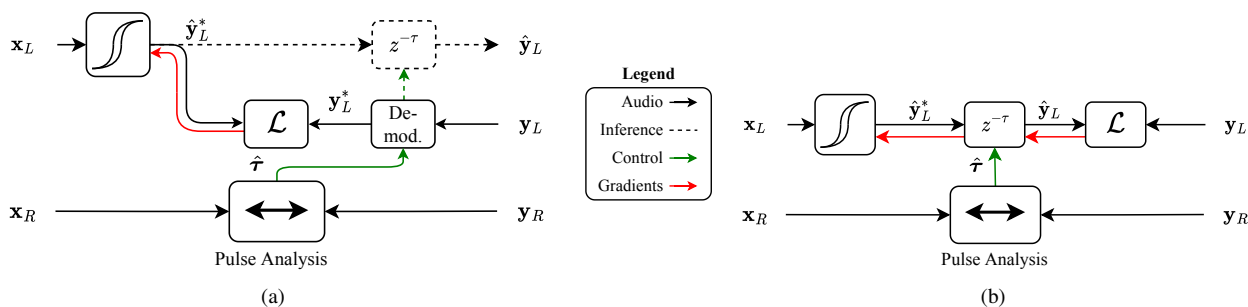
Figure 6: *Supervised approaches for training the nonlinearity: (a) Demodulation and (b) differentiable delay line.*

## 4. DATA COLLECTION

This section provides details concerning the data used in the experimental procedure, including the compiled datasets. The compiled datasets are made available in the accompanying webpage [1].

As input data, we use a fraction of the inputs from Signal-Train [17] for training the nonlinear block. The dataset consists of short musical passages representing varying genres played using various instruments, together with synthetic measurement signals, sampled at $44.1\,\text{kHz}$. A total of $60\,\text{min}$, $20\,\text{min}$, and $15\,\text{min}$ of audio is used for training, validation, and testing, respectively.

### 4.1. Toy Data

We test the modeling architecture using synthetic data, generated via wrapping the VST instance of CHOWTape [7], a white-box modeled tape machine, using Pedalboard [2]. During the generation, the VST instance is set to $16\times$ oversampling using an 8-iteration Newton-Rhapson solver for the tape hysteresis ODE, which are the highest quality settings available. To make sure the virtual tape is sufficiently saturated, the tape drive, tape saturation, and tape bias are set to $(0.75, 0.75, 0.0) \in [0, 1]$, respectively. The timing parameters—the flutter depth, wow depth, and wow variance—are set to $(0.75, 0.75, 1.0) \in [0, 1]$, respectively. We turn off any additional processing from the VST which did not appear in the original research article. Two datasets are collected for the experiments with the toy data: one with only the tape effects enabled and the timing effects disabled, and one with both of the effects enabled.

### 4.2. Real Data

The real data for evaluating the modeling architecture was collected using an Akai 4000D open-reel tape recorder (Fig. 1). The device is a $\frac{1}{4}$ inch, four-track, three-head, stereo recorder from the 1970s, capable of running at $3\frac{3}{4}$ and $7\frac{1}{2}$ inches per second (IPS) and using discrete transistor circuitry for the input/output amplifiers. The data was collected using two types of magnetic tape: a Maxell low-noise/high-output tape and a Scotch low-noise tape from the 1970s. An RME Fireface UCX audio interface was used for recording and playback.

Using a three-head recorder allows for simultaneous recording and playback to and from the tape, allowing the fluctuating time

---

[1] http://research.spa.aalto.fi/publications/ papers/dafx23-neural-tape/

[2] https://github.com/spotify/pedalboard

---

delay between the record and playback heads to be captured. In practice, we connect a stereo line feed from the audio interface into the line inputs of the tape recorder, set monitoring to *TAPE*, and record both the stereo line feed from the tape recorder, as well as a loopback signal from the interface outputs back to its inputs. The recording level of the device was set such that when monitoring the signal entering the tape (monitoring set to *INPUT*), a $0\,\text{dBFS}$ signal from the interface is just below the clipping threshold of the record and playback amplifiers.

The collected data is divided into datasets used for training the lumped nonlinearities, the delay trajectory generator, and the noise generator. We collected two versions of each dataset using the (tape branch, tape speed) configurations (MAXELL, $7\frac{1}{2}$ IPS) and (SCOTCH, $3\frac{3}{4}$ IPS). While initially the same datasets were intended to be used for training both the nonlinear block and the delay trajectory generator, studying the extracted trajectories from the lumped nonlinearity datasets revealed that the accuracy of the trajectory generator would be severely limited by the considered sampling rate of $44.1\,\text{kHz}$, and thus we collected separate high-resolution datasets at $192\,\text{kHz}$ for the trajectory generator, consisting of the same upsampled audio. Finally, to train the noise generator, two datasets consisting of only the hiss captured from the line feed of the tape recorder were collected using the original sampling rate of $44.1\,\text{kHz}$.

## 5. IMPLEMENTATION DETAILS

This section provides details concerning implementing the different components in the modeling architecture, including modeling training and the loss functions used.

### 5.1. Supervised Approaches

The first two approaches use supervised training to optimize the weights of the nonlinear block, aligning the target and output segments using the proposed methods shown in Fig. 6. The RNN is trained using truncated back-propagation through time (TBPTT) [18], allowing the RNN state to initialize before tracking the gradients. For the first approach, the RNN state is initialized for 1024 steps, and for the second, the initialization length is determined by taking the next power of two of the maximum delay length in samples encountered in the training dataset. We use a hidden size of 64 as preliminary experiments indicated that increasing the hidden size beyond this did not bring an improvement in model performance. We use Adam with the default hyperparameters as implemented in PyTorch as the optimizer. We use a learning rate

of $1 \times 10^{-3}$ and reduce it with a factor of 0.75 every time the validation loss has not improved for 10 epochs. A batch size of 32 is used for all the experiments and the models are trained using a graphical processing unit for 4 hours.

To compute the prediction discrepancy against the target output for the supervised approaches, the error-to-signal ratio (ESR) loss is used [19]. Details concerning the loss function can be found in the paper by Damskägg et al. [19].

### 5.2. Adversarial Approach

For the adversarial training method, TBPTT was also used. As the time-varying delay line is initially filled with zeros, the initialization stage is run for a number of steps equal to the maximum measured delay-line length. This ensures that the delay line is filled with real values during TBPTT. A segment length of 2 seconds was used during training, with parameter updates being carried out every 16384 samples. The longer TBPTT length was used as the discriminator model uses a time-frequency representation of the signal as input. As such, longer input lengths increase the frequency resolution that is seen by the discriminator model.

The discriminator and RNN model are alternately trained using the hinge loss function described by Kumar et al. [20]. A batch size of 16 was used during training. During validation, paired data was used to evaluate the RNN model, with the measured delay line being applied at the output of the RNN. Training was run for 50 epochs, with a multi-resolution log spectral magnitude loss being used to select the best performing model weights.

### 5.3. Noise Generator

We train our diffusion models following the recommendations by Karras et al. [15]. Considering that the standard deviation of the recorded data is, approximately, $\sigma_{\text{data}} = 8 \times 10^{-4}$, the noise schedule is defined so that, during sampling, the Gaussian noise level decreases logarithmically across the reverse diffusion process from $\sigma_{\max} = 0.1$ (completely masking the data) to $\sigma_{\min} = 5 \times 10^{-5}$ (perceptually insignificant). During training, the model is trained with the L2 preconditioned objective from [15], where the noise level is sampled randomly with a LogUniform distribution. The model is trained with the Adam optimizer with the default momentum hyperparameters and a learning rate of $2 \times 10^{-4}$. An exponential moving average of the weights with a decay factor of 0.999 is tracked during training and used as the final inference model.

A standard time-domain convolutional U-Net is used as the backbone deep neural network architecture, which is conditioned on a noise level embedding that allows weights to be shared across different noise levels. The total number of parameters adds to 127k, which is relatively low when compared to standard practice for diffusion models, but has proven qualitatively to be enough for this particular use-case. Inference is performed with a denoising diffusion implicit model (DDIM) [21] sampler with a subtle amount of stochasticity, which allows for a trade-off between sampling speed and quality by adjusting the number of sampling steps. We use 16 steps in our experiments, but we observe that this number can be reduced down to 6 without a significant quality loss.

The noise generator model is trained with samples of 1.5-s at the sampling rate of 44.1 kHz. However, given that the architecture is fully-convolutional, the segment size could be freely adapted during inference. In addition, arbitrarily long sequences can be generated by applying a chunked autoregressive sampling strategy,
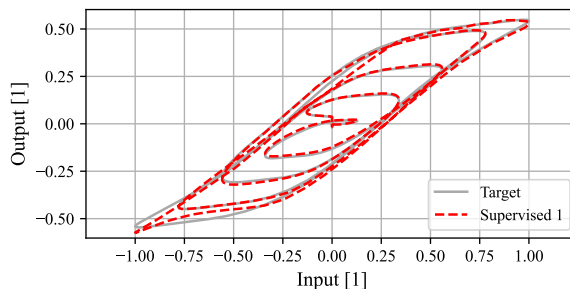


Figure 7: *Model hysteresis using toy data - Lumped nonlinearities only.*

where separate frames are concatenated and inter-frame coherence can be ensured by applying a zero-shot outpainting technique [22].

### 5.4. Trajectory Generator

The delay trajectory generator is trained in a very similar way as the noise generator, specified in Sec. 5.3, while only some implementation details concerning the characteristics of the data differ. The model is trained with 5.2-s segments at a sampling frequency of 100 Hz, defined by the measuring pulse frequency (see Sec. 3.1), resulting on segments of 512 samples. During training, every delay trajectory segment is mean-normalized, leaving only the local fluctuations from the average delay as the distribution to be modeled. The backbone neural network architecture is also a standard convolutional U-Net with 77k trainable parameters; we refer to the source code for further details. The noise schedule design is motivated analogous to Sec. 5.3 and depends on the statistics of the dataset. For the toy data experiment (see Sec. 6), the approximated data standard deviation is $\sigma_{\text{data}} = 6.8 \times 10^{-3}$, and the noise schedule is designed between $\sigma_{\max} = 0.5$ and $\sigma_{\min} = 1 \times 10^{-5}$. For the real data experiment (see Sec. 7), the data standard deviation is approximately $\sigma_{\text{data}} = 1 \times 10^{-4}$, and we found that $\sigma_{\max} = 0.01$ and $\sigma_{\min} = 1 \times 10^{-5}$ were a suitable design choice. The sampling is performed with a 10-step DDIM sampler [21], but we observed that the number of discretization steps could be reduced down to only 4 with minimal qualitative differences.

### 6. EXPERIMENT 1: TOY DATA

This section presents experiments using synthetic data in two subsections. Sec. 6.1 studies the capability of the modeling method for the hysteretic magnetization of the tape without the fluctuating timing effects. Later, the timing effects are also enabled, allowing the evaluation of the three training schemes for the nonlinearity in Sec. 6.2 and the trajectory generator in Sec. 6.3. The toy data does not contain a noise component, however. Audio examples for these experiments are available on the accompanying web page [1].

### 6.1. Lumped Nonlinearities Only

To evaluate model performance, we use the ramped sine technique for the hysteresis [23]. Additionally, we encourage readers to listen to the example predictions on the accompanying web page[1]. The ramped sine technique is especially useful here, since it evaluates both the deadzone effect resulting from an under-biased tape and the saturation at higher amplitudes.

The model hysteresis versus the target is shown in Fig. 7. While the match is not perfect, the model clearly learns the shape
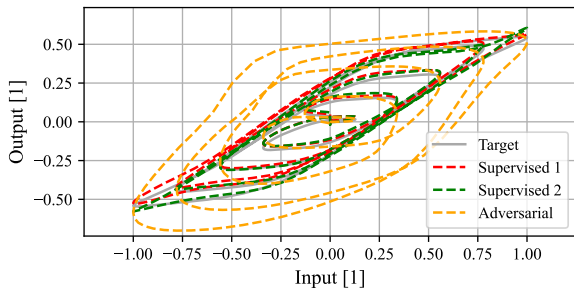
Figure 8: *Model hysteresis using toy data - Nonlinearities and timing effects.*



Figure 9: *Measured (left) and generated (right) delay trajectories using toy data.*

of the hysteresis loop, including the deadzone effect, the saturation of the tape, as well as the loop width. Listening and comparing the model predictions against the target further validates this finding, confirming the suitability of the RNN architecture for modeling the type of nonlinearity.

### 6.2. Lumped Nonlinearities and Timing Effects

The various training schemes are evaluated as in Sec. 6.1, on top of which the model losses over the test set are compared. Since the adversarial models are not trained with a time-domain loss, we include a multi-resolution short-time Fourier transform (STFT) loss [24] in the comparison. We use the default hyperparameters for the method as implemented in the Auraloss library [25]. Model predictions can be found in the accompanying webpage[1].

The hysteresis of the models trained using the three approaches versus the target is shown in Fig. 8. While it can be seen that the models trained using the supervised approaches match the shape of the hysteresis well, the hysteresis shape of the adversarially trained model deviates from the target. This can be explained by the adversarial model being trained on a time-frequency domain loss, which does not enforce strict time-domain matching, which is the case for the two supervised models.

The $\mathcal{L}_{\text{ESR}}$ and $\mathcal{L}_{\text{STFT}}$ losses over the test set are listed in Table 1, where the best (smallest) results are highlighted with bold font. The losses for the first supervised approach are computed either via using the real delay trajectories to demodulate the targets (*Demodulated*, similar to training) or applying the trajectories to the predictions (*Delayed*, similar to inference). As can be seen from the results, the second supervised approach produces the best overall performance across the considered metrics. While the time-domain loss $\mathcal{L}_{\text{ESR}}$ is two orders of magnitude higher for the adversarial approach than for the supervised approaches, their time-frequency domain losses $\mathcal{L}_{\text{STFT}}$ are of similar magnitude. For the first supervised approach, it can be seen that for both considered losses, the delayed computational method results in a smaller error. This finding suggests that demodulating the outputs can produce a larger error in comparison to delaying them with an interpolated delay line.

Table 1: *Toy Data: Nonlinearities and timing effects.*

| Approach | $\mathcal{L}_{\text{ESR}}$ | | $\mathcal{L}_{\text{STFT}}$ | |
| | Demod. | Delayed | Demod. | Delayed |
| --- | --- | --- | --- | --- |
| Supervised I | 0.031 | **0.029** | 0.645 | 0.536 |
| Supervised II | – | **0.029** | – | **0.488** |
| Adversarial | – | 1.567 | – | 1.772 |
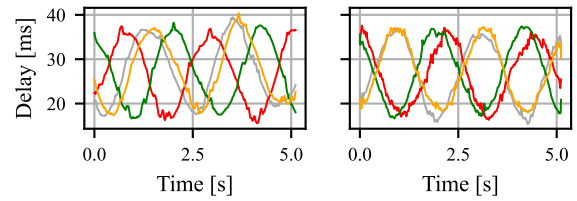
### 6.3. Trajectory Generator

We experiment with the diffusion model approach to generate the delay trajectories from the toy experiment, which have been synthesized as described in Sec. 4.1. Despite our best efforts to devise a methodology for objective evaluation, we found that the stochastic nature of the data and the complexities involved made it difficult to quantify its effectiveness in a reliable and consistent manner. As such, we rely on a merely qualitative assessment, which we believe provides sufficient evidence of the successful model capabilities. In Fig. 9, we show a qualitative comparison between measured trajectories and generated ones. In this case, 10 iterative steps were used to sample from the diffusion model, requiring 10 function evaluations of the neural network. The data contains a prominent sinusoidal component with some spurious artifacts, which seem to be accurately modeled by the diffusion model.

## 7. EXPERIMENT 2: REAL DATA

Next, the proposed method is evaluated using real data collected from the Akai 4000D tape recorder. Since the toy data did not include a noise component, this section serves as the first validation for the capability of the noise generator to learn the character of the media, as well as further validates the performance of the other two architectural components. Audio examples from the conducted experiments can be found on the web page[1].

### 7.1. Lumped Nonlinearities and Timing Effects

We start with the same evaluation strategy as in Sec. 6.2, but find that the magnetic field produced by the recording head is not sufficient to saturate the considered tape formulations, as has been encountered earlier [1]. Thus, instead of comparing the hysteresis of the trained models, we focus on the learned magnitude response and nonlinear distortion components [26]. Only one configuration (MAXELL $7\frac{1}{2}$ IPS) is evaluated here for the sake of brevity. The model predictions can be found on the accompanying web page[1].

We find the responses of the supervised models similar, and only show the learned magnitude response and nonlinear distortion components versus the target for the first supervised approach in Fig. 10a. As can be seen, the model closely matches the shape of the linear response: the attenuated middle frequencies, the subtle emphasis of the highs, as well as the high and low-frequency roll-offs. While the target also portrays the head-bump effect in the low frequencies, this aspect is not matched by the model. We suspect that this might have to do with the dataset used for training the models not having enough low-frequency content to sufficiently learn this frequency band. While the model also learns to produce nonlinear distortion from the target data, the shape of the contours is not matched well, and the model starts to alias above
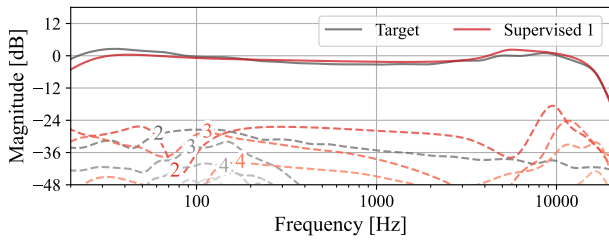
(a) *Supervised I.*
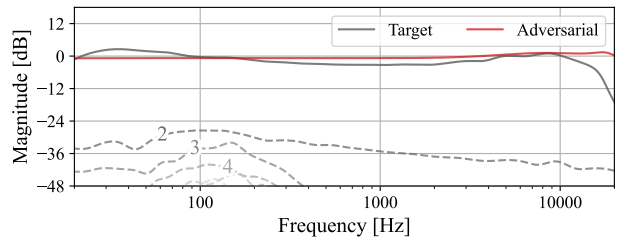


(b) *Adversarial.*

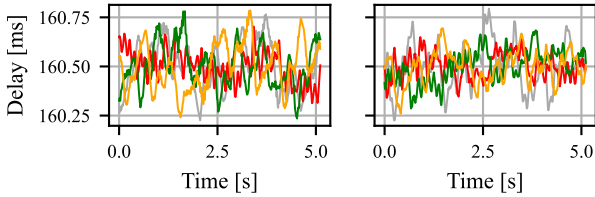Figure 10: *Model magnitude responses (solid) and distortion components (dashed), MAXELL $7\frac{1}{2}$ IPS.*



Figure 11: *Measured (left) and generated (right) delay trajectories using real data.*



Figure 12: *Average spectra and standard deviations of delay trajectories using real data.*

about 5 kHz, as seen in the sudden increase of nonlinear distortion components in Fig. 10a. The learned magnitude response and nonlinear distortion components for the adversarially trained model are shown in Fig. 10b. Now the match is poor in both the linear response as well as the nonlinear harmonic components.

The $\mathcal{L}_{\text{ESR}}$ and $\mathcal{L}_{\text{STFT}}$ losses over the test set for all of the approaches are listed in Table 2. Similarly as in Sec. 6.2, the two supervised approaches outperform the adversarial approach in terms of both of the considered metrics, with an order of magnitude difference in the time-domain $\mathcal{L}_{\text{ESR}}$ loss. Unlike before, the first supervised approach performs slightly better than the second approach, although the difference is not large. While the two methods for computing the losses for the first supervised approach produce similar evaluation metrics, this time the demodulated computational method brings slight improvements in the $\mathcal{L}_{\text{STFT}}$ loss. This finding suggests that the error produced by the two computational methods also depends on the type of data used.

### 7.2. Trajectory Generator

Fig. 11 shows delay trajectory samples from the measured data compared to those generated with a 10-step diffusion model. The generated delay trajectory waveforms look realistic at first glance but, in order to provide more insights into the model behavior, we conduct a spectral analysis. Fig. 12 shows the summary statistics (mean and standard deviation) of the long-term spectrum of the measured trajectories compared to that of a batch of 256 generated samples. The data presents some characteristic spectral peaks that the diffusion model is succeeding at replicating. It can also
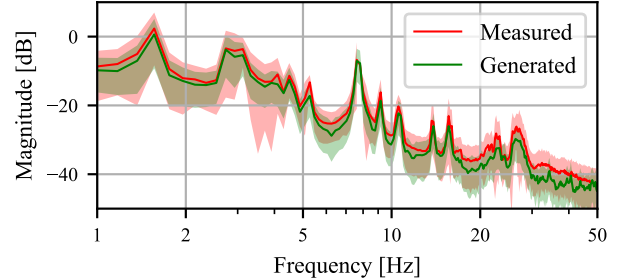
Table 2: *Real Data: Nonlinearities and timing effects.*

| | $\mathcal{L}_{\text{ESR}}$ | | $\mathcal{L}_{\text{STFT}}$ | |
| Approach | Demod. | Delayed | Demod. | Delayed |
|---|---|---|---|---|
| Supervised I | 0.066 | **0.065** | **1.597** | 1.649 |
| Supervised II | – | 0.092 | – | 1.971 |
| Adversarial | – | 1.193 | – | 2.437 |

be observed that the average spectral magnitude of the generated trajectories is slightly lower than the target; we attribute this to the over-denoising phenomena that most diffusion models show, as it was observed in [15]. Nevertheless, we believe that these minor dissimilarities will pose no perceptual difference.

### 7.3. Noise Generator

We assess the diffusion noise generator qualitatively, similar to the delay trajectory generator, since an objective evaluation is not feasible. Fig. 13 displays the long-term spectra of the noise data and compares it with that of the generated noises using the diffusion model with 16 discretization steps. The plot in Fig. 13 has been smoothed using a 1/6th octave band. The majority of the energy in the target data distribution is concentrated in the low-frequency region, with some spectral peaks caused by electrical noise. The diffusion model successfully replicates the spectral distribution, and the over-denoising effect observed in Sec. 7.2 does not occur as a consequence of using a stochastic sampler. Additionally, the noise generator can effectively model some non-stationary local characteristics in the noise data that are not adequately represented in the spectral analysis. We refer the reader to the audio examples on the companion webpage[1].

## 8. CONCLUSIONS

This work proposed an architecture for modeling the character of magnetic tape recorders, consisting of three components for reproducing the different aspects of the target: 1) a nonlinear block for the joint effects of the magnetic recording process as well as the record and playback amplifiers, 2) a time-varying delay line controlled by a delay trajectory generator for imperfections in the tape transport, and 3) a noise generator for the tape hiss. The different blocks were implemented using separate neural network archi-
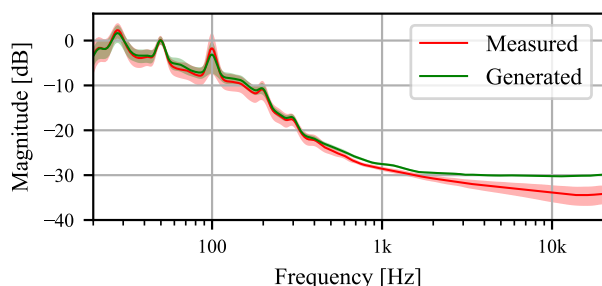
Figure 13: *Average spectra and standard deviations of tape hiss.*

tectures: an RNN for the nonlinear block and separate diffusion models for the delay and noise generators. Three training schemes were considered for the nonlinear block: two supervised and one adversarial.

Our results indicate that the RNN architecture is suitable for learning the characteristic hysteretic nonlinear behavior of the tape magnetization. This was also found recently elsewhere for the related case of audio transformers [4]. Based on objective and qualitative evaluation and informal listening, the two supervised approaches for the nonlinear block together with the generative models for the delay trajectories and tape hiss capture the perceptual character of the tape recorder as a whole. While the proposed supervised training schemes require the target machine to operate in stereo, an aspect of which was circumvented by the adversarial approach, this latter approach did not prove mature yet in learning the nonlinear character of the tape.

## 9. ACKNOWLEDGMENTS

## 10. REFERENCES

[1] V. Välimäki, S. Bilbao, J. O. Smith, J. S. Abel, J. Pakarinen, and D. Berners, *DAFX: Digital Audio Effects*, chapter "Virtual analog effects", pp. 473–522, John Wiley & Sons, Ltd, Chichester, UK, Mar. 2011.

[2] A. Wright, E.-P. Damskägg, and V. Välimäki, "Real-time black-box modelling with recurrent neural networks," in *Proc. Int. Conf. Digital Audio Effects (DAFX)*, Birmingham, UK, Sept. 2019, pp. 173–180.

[3] E. Moliner and V. Välimäki, "Realistic gramophone noise synthesis using a diffusion model," in *Proc. Int. Conf. Digital Audio Effects (DAFX)*, Vienna, Austria, Sept. 2022, pp. 240–247.

[4] O. Massi, A. I. Mezza, R. Giampiccolo, and A. Bernardini, "Deep learning-based wave digital modeling of rate-dependent hysteretic nonlinearities for virtual analog applications," *EURASIP J. Audio Speech Music Process.*, vol. 2023, no. 1, Mar. 2023.

[5] H. N. Bertram, *Theory of Magnetic Recording*, Cambridge University Press, Cambridge, New York, 1994.

[6] M. Camras, *Magnetic Recording Handbook*, Springer Netherlands, Dordrecht, Netherlands, 1998.

[7] J. Chowdhury, "Real-time physical modelling for analog tape machines," in *Proc. Int. Conf. Digital Audio Effects (DAFX)*, Birmingham, UK, Sept. 2019, pp. 292–298.

[8] S. Arnardottir, J. S. Abel, and J. O. Smith, "A digital model of the Echoplex tape delay," in *Proc. Audio Eng. Soc. 125th Conv.*, San Francisco, CA, Oct. 2008.

[9] J. Kaloinen, "Neural modeling of the audio tape echo effect," M.S. thesis, Aalto University, Espoo, Finland, Aug. 2022.

[10] E. D. Daniel, "Tape noise in audio recording," *J. Audio Eng. Soc.*, vol. 20, no. 2, pp. 92–99, Mar. 1972.

[11] U. Zölzer, *DAFX: Digital Audio Effects*, John Wiley & Sons Ltd, Chichester, UK, second edition, 2011.

[12] J. Engel, L. H. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in *Proc. ICLR*, Addis Ababa, Ethiopia, Aug. 2020.

[13] A. Wright, V. Välimäki, and L. Juvela, "Adversarial guitar amplifier modelling with unpaired data," in *Proc. IEEE ICASSP*, Rhodes Island, Greece, June 2023.

[14] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. NeurIPS*, Online, Dec. 2020, vol. 33, pp. 6840–6851.

[15] T. Karras, M. Aittala, T. Aila, and S. Laine, "Elucidating the design space of diffusion-based generative models," in *Proc. NeurIPS*, New Orleans, LA, Oct. 2022, vol. 35, pp. 26565–26577.

[16] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *Proc. ICLR*, Vienna, Austria, May 2021.

[17] S. Hawley, B. Colburn, and S. I. Mimilakis, "Profiling audio compressors with deep neural networks," in *Proc. Audio Eng. Soc. 147th Conv.*, New York, NY, Oct. 2019.

[18] J. L. Elman, "Finding structure in time," *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, 1990.

[19] E.-P. Damskägg, L. Juvela, E. Thuillier, and V. Välimäki, "Deep learning for tube amplifier emulation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Brighton, UK, May 2019, pp. 471–475.

[20] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, *et al.*, "MelGAN," in *Proc. NeurIPS*, Vancouver, Canada, Dec. 2019, vol. 32, pp. 14843–14854.

[21] J. Song, C. Meng, and S. Ermon, "Denoising diffusion implicit models," in *Proc. ICLR*, Vienna, Austria, May 2021.

[22] J. Ho, T. Salimans, A. Gritsenko, W. Chan, M. Norouzi, and D. J. Fleet, "Video diffusion models," in *Proc. NeurIPS*, New Orleans, LA, Dec. 2022, vol. 35, pp. 8633–8646.

[23] M. Holters and U. Zölzer, "Circuit simulation with inductors and transformers based on the Jiles-Atherton model of magnetization," in *Proc. Int. Conf. Digital Audio Effects (DAFx)*, Brno, Czech Republic, Sept. 2016, pp. 55–60.

[24] R. Yamamoto, E. Song, and J.-M. Kim, "Parallel Wave-GAN," in *Proc. IEEE ICASSP*, Barcelona, Spain, May 2020, pp. 6199–6203.

[25] C. J. Steinmetz and J. D. Reiss, "Auraloss: Audio-focused loss functions in PyTorch," in *Proc. DMRN+15*, London, UK, Dec. 2020.

[26] A. Farina, "Simultaneous measurement of impulse response and distortion with a swept-sine technique," in *Proc. Audio Eng. Soc. 108th Conv.*, Paris, France, Feb. 2000.