OPEN ACCESS Freely available online
L. Renault, R. Mignot and A. Roebel,
"DDSP-Piano: A Neural Sound Synthesizer
Informed by Instrument Knowledge"
J. Audio Eng. Soc., vol. 71, no. 9, pp. 552–565, (2023 September).
DOI: https://doi.org/10.17743/jaes.2022.0102

DDSP-Piano: A Neural Sound Synthesizer Informed by Instrument Knowledge

LENNY RENAULT, RÉMI MIGNOT, AND AXEL ROEBEL
(lenny.renault@ircam.fr) (remi.mignot@ircam.fr) (axel.roebel@ircam.fr)

STMS - UMR9912, IRCAM, Sorbonne Université, CNRS, Ministère de la Culture, Paris, France

Instrument sound synthesis using deep neural networks has received numerous improvements over the last couple of years. Among them, the Differentiable Digital Signal Processing (DDSP) framework has modernized the spectral modeling paradigm by including signal-based synthesizers and effects into fully differentiable architectures. The present work extends the applications of DDSP to the task of polyphonic sound synthesis, with the proposal of a differentiable piano synthesizer conditioned on MIDI inputs. The model architecture is motivated by high-level acoustic modeling knowledge of the instrument, which, along with the sound structure priors inherent to the DDSP components, makes for a lightweight, interpretable, and realistic-sounding piano model. A subjective listening test has revealed that the proposed approach achieves better sound quality than a state-of-the-art neural-based piano synthesizer, but physical-modeling-based models still hold the best quality. Leveraging its interpretability and modularity, a qualitative analysis of the model behavior was also conducted: it highlights where additional modeling knowledge and optimization procedures could be inserted in order to improve the synthesis quality and the manipulation of sound properties. Eventually, the proposed differentiable synthesizer can be further used with other deep learning models for alternative musical tasks handling polyphonic audio and symbolic data.

0 INTRODUCTION

Digital instruments and synthesizers have greatly impacted the way music is being composed, produced, and played and have thus participated in the shaping of new musical genres. Analog and digital synthesizers undoubtedly allowed for the exploration of new sounds, by producing them in very different manners compared to physical instruments. Additionally, progresses made in instrument modeling have made it possible for more musicians to use the sounds of more acoustic instruments in a simplified way. Therefore, for these instrument models to be effective in the music-making process, they have to be easily controlled while accurately reproducing the subtle nuances of the modeled instrument sounds.

Among the many methods for digital sound synthesis, generative models based on neural networks have gained a consequent interest recently, as they have shown to be capable of reproducing, manipulating, and understanding musical sounds in unprecedented ways. Their ability to model nonlinear relationships, as well as the release of associated datasets, have allowed the flourishing of neural-based methods for instrument sound synthesis. For monophonic and percussive sound synthesis, such models include autoregressive models [1], recurrent models [2], (variational) auto-encoders [3, 4], *Generative Adversarial Network* (GAN) [5–7], and diffusion models [8–10]. Although these neural methods generate or manipulate audio in the waveform or a time-frequency domain, the *Differentiable Digital Signal Processing* (DDSP) framework [11], on the other hand, predicts controls for manipulating audio synthesizers. By implementing traditional synthesizers and digital processing operations as differentiable layers, the spectral modeling paradigm [12] has been made compatible and controllable with neural networks. These DDSP components are designed to exhibit and to take advantage of known properties of audio signals, such as periodicity and harmonicity. As these strong priors on the sound structure are introduced, the amount of training data and model parameters can be significantly reduced, and the synthesis process is more interpretable.

In view of these results, new DDSP components were developed and used in end-to-end neural models by revisiting synthesizer and signal-based techniques. For sound generation, such components include wavetables [13], waveshapers [14], and frequency modulation [15]. As for differentiable signal processing, infinite impulse response filters [16], parametric equalizers [17], and artificial reverbera-

tion [18] were also revamped. For better manipulation of DDSP-based synthesis models, several improvements [19–21] have been made to make them controllable with the *Musical Instrument Digital Interface* (MIDI) protocol. Finally, polyphonic mixtures of audio signals were also achieved by combining multiple monophonic DDSP-based models [22, 23].

As of now, all these methods have only been applied for modeling the sound of monophonic audio sources (instrument or voice). Although DDSP does not prevent its application for the polyphonic case, to the best of the authors' knowledge, no extension has been made to the DDSP components for handling simultaneous pitches on a single instrument. Specifically, the piano has been one of the most popular instruments through the history of western music, notably because of its versatility as a polyphonic instrument with a wide range of pitches and dynamics. The controls for playing it remain fairly simple despite its complexity, which inspired researchers for modeling it with different simulation systems for decades before the advent of deep generative models.

This paper presents DDSP-Piano, a MIDI-controllable model that extends the DDSP approach for handling polyphonic inputs. Through the modularity of the DDSP components, the model incorporates high-level modeling knowledge as structural constraints in order to reproduce specificities of the piano sound (e.g., partials inharmonicity and beatings). This fully differentiable synthesizer was trained on a publicly available dataset of piano performances and was evaluated in a listening test against other piano sound synthesis models. Ratings indicate that with significantly fewer parameters, and even with less training data, the DDSP-Piano model has a better sound quality than a state-of-the-art neural synthesizer. Deeper analysis were conducted on the system initially presented in [24], which reveal the behaviors learned by the model and help identifying its shortcomings and how it can be further improved.

This paper is organized as follows. SEC. 1 presents different categories of piano sound synthesizers. Our proposed model is then introduced in SEC. 2, with its implementation and training procedure explained in SEC. 3. Quality evaluation of the model is then conducted against other piano sound generation algorithms in SEC. 4, and SEC. 5 shows qualitative results of the synthesis made by the DDSP-Piano. Broader reflections on the approach are given in SEC. 6. Completing this paper, examples of audio synthesis and the source-code for model training and inference can be found online.[1]

1 RELATED WORKS

Numerous methods have been proposed for piano sound synthesis, each with varying complexity, needs for data, and overall quality. The most common approach in the industry, which is also the most straightforward, is the concatenative synthesis, or sampling-based synthesis [25]. High-fidelity recordings of isolated notes are played back upon triggers from the MIDI input. The notes can be recorded at different velocities to cover the amplitude range, which can require significant memory storage, depending on the chosen resolution. Although single notes can be perceived as realistic, mutual interactions between simultaneous notes (such as sympathetic resonances) are not reproduced with this approach, and the user has very limited control over the model.

On the contrary, physical-modeling-based systems rely on explicitly modeling the sound generation and propagation processes in the physical instrument. These systems can achieve realistic, interpretable, and controllable sound synthesis, but they require extensive modeling and precise measurements of physical elements [26]. For practical usages, such approaches can be efficiently implemented with digital waveguides [27] or modal synthesis [28].

Signal-based methods can model the instrument by analysis and synthesis of audio examples with hand-crafted models. Underlying models include additive synthesis [29] and source-filter models [30]. These lightweight approaches are controllable and flexible as they can be directly applied to other instruments, but they often lack realism in the synthesis because of insufficient representation of physical details of the instrument or too-simplistic controls.

Finally, data-driven neural-based systems train black-box models to synthesize audio from a large annotated dataset. Most of these models adapt successful text-to-speech techniques to the task of MIDI-to-audio synthesis for piano. The first works for this category of systems synthesize audio directly from the MIDI data using an auto-regressive WaveNet [1]. Others works make use of an acoustic model followed by a vocoder model to synthesize audio while usually predicting Mel-spectrograms as the intermediate audio representation [31–33, 10]. The authors from [34] achieved better quality by predicting MIDI-filter-bank–based spectra instead: this time-frequency representation is a variant of the Mel-spectrogram, in which the filters for computing it from the *Short-Term Fourier Transform* (STFT) are centered around the MIDI note frequencies instead of the Mel frequencies. Although differentiable, these neural-based systems require a significant amount of annotated recordings [1], and they do not explicitly model instrument properties. Controlling these systems is limited to the conditioning inputs provided during model training.

2 MODEL ARCHITECTURE

The proposed synthesis model is a harmonic-plus-noise synthesizer [12] with polyphonic controls and outputs. It separately generates the inharmonic and noisy components $y^{additive}$ and y^{noise} of up to P simultaneous notes. The synthesized audio $\hat{y}$ is produced by summing all monophonic signals and by applying the estimated response IR_i of the recording environment i:

$$\hat{y}(t) = \left(\mathrm{IR}_i * \sum_{p=1}^{P} \left(y_p^{additive} + y_p^{noise} \right) \right)[t]. \tag{1}$$

[1] https://github.com/lrenault/ddsp-piano

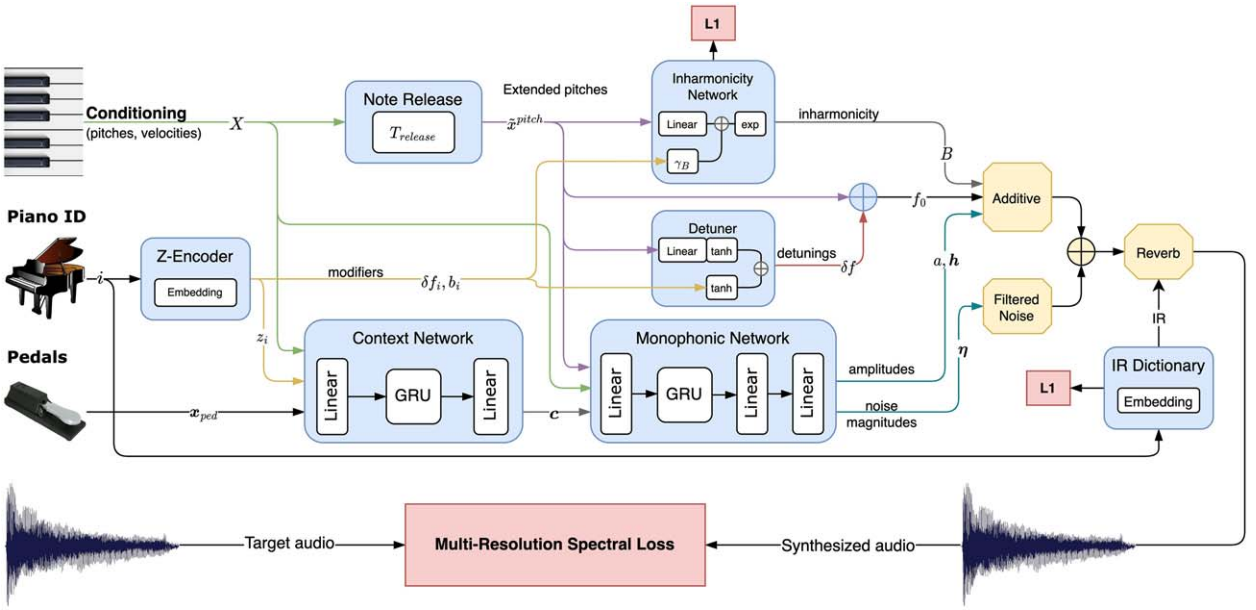

Fig. 1. Full architecture of the proposed piano sound synthesizer. The rounded boxes represent the trained modules for the control of the synthesis. The synthesis modules from DDSP are represented by octagon boxes (*Additive, Filtered Noise*, and *Reverberation*). Finally, the *Multi-Resolution Spectral Loss* compares the input *target signal* (bottom left) and the output *synthesized sound* (bottom right).

The following sections detail the submodules composing the full model architecture, which is illustrated in Fig. 1.

2.1 Inputs

DDSP-Piano is conditioned on all the controls a pianist has over its instrument: the sequence of notes being played, the pedals action, and the recording environment. The encodings chosen for each control are detailed in the following.

The monophonic conditioning of DDSP [11, 21] is composed of F0 and loudness control signals, which provides the instantaneous fundamental frequency and intensity of a note sequence at a constant frame rate. For MIDI compatibility, the F0 control is substituted by an active pitch control signal $x^{pitch}(t)$ that indicates the MIDI pitch of the note at time t, taking the sustain pedal effect into account for its duration. Note that $x^{pitch}(t) = 0$ means that no note is currently being played at frame t. Likewise, the loudness control is replaced by an onset velocity control $x^{vel}(t)$ that specifies the note velocity (scaled to the range [0, 1]) only at onset time. This encoding allows for the disentangling of sustained notes from repeated notes within an active sustain pedal, as in [31]. The polyphonic conditioning $X(t)$ is obtained by splitting all the input notes into P sequences of nonoverlapping notes, or *voices*, and by stacking these monophonic conditionings as in [22]:

$$X(t) = \{x_p^{pitch}(t), x_p^{vel}(t)\}_{p \leq P}. \tag{2}$$

The pedal input controls $\boldsymbol{x}^{ped}(t)$ are extracted from the MIDI data at the same frame rate as the conditioning input $X(t)$. Most notably, the sustain pedal control is included.

Finally, the piano model, the room reverberation, and the microphone choice and placement are all entangled independently of the piano performance: each recording environment is provided as a one-hot encoding $i \in [1, I]$, for I different recording environments in the dataset.

2.2 Global Model

One piano model can differ from another one because of its size and its tuning, which changes its inharmonicity profile over the piano pitch range, or *tessitura* [35], and its global detuning. We thus use an embedding layer, the *Z-Encoder*, to compute an embedding vector $\boldsymbol{z}_i$, an inharmonicity modifier b_i, and an instrument specific detuning δf_i for each recording environment i.

Also, during a performance, the pedals activity and the interaction between simultaneous notes (e.g., sympathetic resonances) can change the timbre of an individual note [26]. This effect is modeled by a C-dimensional context control $\boldsymbol{c}(t)$ computed by the *Recurrent Neural Network* (RNN)-based *context network* $\mathcal{F}$, from the piano embedding $\boldsymbol{z}_i$, the pedal controls $\boldsymbol{x}^{ped}(t)$ and the conditioning $X(t)$:

$$\boldsymbol{c}(t) = \mathcal{F}\{X(\tau), \boldsymbol{x}^{ped}(\tau), \boldsymbol{z}_i\}_{\tau \leq t}. \tag{3}$$

This context control is duplicated across all monophonic voices, which give the subsequent monophonic layers (to be presented in SEC. 2.3) access to global and polyphonic information and thus adjust the computation of monophonic note properties accordingly.

2.3 Monophonic String Model

The piano dampers stop the strings from vibrating when the key is released. However, the attenuation of the energy is not instantaneous, and higher notes do not even have dampers [26]. Hence, in practice, the piano strings still vibrate for a certain amount of time after the note offset. Taking inspiration from the *release* parameter of digital

synthesizers, a *Note Release* module is implemented to generate an extended pitch signal $\tilde{x}^{pitch}(t)$ by prolonging the active pitch component of the conditioning signal $x^{pitch}(t)$ by a learned duration $T_{release}$. Note that the extended pitch conditioning signal $\tilde{x}^{pitch}(t)$ does not replace the original pitch conditioning $x^{pitch}(t)$, as we would lose the note offset information.

Furthermore, the stiffness of piano strings induces the partials of a piano note to not be pure harmonics of the fundamental frequency. This characteristic is implemented with an explicit *inharmonicity model* over the piano tessitura, taken from [35]: the inharmoncity factor along the p-th voice $B_p(t)$ is computed from the extended pitch $\tilde{x}_p^{pitch}(t)$ and the instrument specific modifier b_i:

$$B_p(t) = \exp(\alpha_T \tilde{x}_p^{pitch}(t) + \beta_T) + \exp(\alpha_B \tilde{x}_p^{pitch}(t) + \beta_B + \gamma_B b_i), \quad (4)$$

with $\{\alpha_T, \beta_T\}$ and $\{\alpha_B, \beta_B\}$ the parameters of the linear asymptote in the treble and bass ranges, respectively. The treble asymptotes are very similar across all pianos, according to [36], so b_i only influences the bass asymptote, weighted by the parameter γ_B.

Another peculiarity of the piano tone is the presence of string duets and triplets: higher strings are duplicated in order to even out the loudness and the duration of notes across the whole tessitura. Partials beating can be heard because the duplicated strings are slightly detuned from one another [37]. A simple approximation to model this phenomenon is to consider a monophonic note as the sum of n_{strings} sub-strings, each detuned by a detuning factor δf. The *detuner* submodel gathers the per-string deviations predicted by a time-distributed linear layer g_δ from the pitch and a global instrument-specific detuning δf_i:

$$\boldsymbol{\delta f}_p(t) = \tanh(g_\delta(\tilde{x}_p^{pitch}(t)) + \tanh(\delta f_i). \quad (5)$$

Each command contributing to the detuning is limited to a semitone range $[-1, 1]$, as in [21], using the tanh activation function.

Finally, the spectral envelopes of notes and their evolution are predicted by the *monophonic network* $\mathcal{G}$. It is implemented as a causal RNN that computes the remaining synthesizers controls from the extended pitch $\tilde{x}^{pitch}(t)$, the conditioning vector $X(t)$, and the context vector $\boldsymbol{c}(t)$. This recurrent network is applied along each voice $p \leqslant P$, in order to learn a monophonic string model and to predict the notes amplitude $a(t)$, the energy distribution $\boldsymbol{h}(t)$ for K partials, and noise filter magnitudes $\boldsymbol{\eta}(t)$:

$$a_p(t), \boldsymbol{h}_p(t), \boldsymbol{\eta}_p(t) = \mathcal{G}\{X_p(\tau), \tilde{x}_p^{pitch}(\tau), \boldsymbol{c}(\tau)\}_{\tau \leq t}. \quad (6)$$

2.4 Differentiable Synthesizers

The outputs of the neural network are used to control the differentiable synthesizers, which generate and process audio signals in the spectral modeling paradigm [12]. As in the original DDSP autoencoder [11], controls are upsampled from the controls frame rate to the audio sampling rate. Amplitude, energy distribution, and noise filter magnitude controls are also scaled with the same modified sigmoid function in order to be non-negative.

Along a monophonic voice $p \leqslant P$, the *additive synthesizer* generates the inharmonic audio component $y_p^{additive}(t)$ of the piano notes. It sums multiple sinusoids at frequencies computed from the extended pitch $\tilde{x}_p^{pitch}(t)$, inharmonicity $B_p(t)$, and detuning $\delta f_p(t)$ controls, and with amplitudes provided by the global amplitude $a_p(t)$ and harmonic distribution $\boldsymbol{h}_p(t)$:

$$y_p^{additive}(t) = \frac{a_p(t)}{n_{\text{strings}}} \sum_{n=1}^{n_{\text{strings}}} \sum_{k=1}^{K} h_{p,k}(t) \sin(\Phi_{p,n,k}(t)), \quad (7)$$

with $\Phi_{p,n,k}(t)$ the instantaneous phase of the k-th partial:

$$\Phi_{p,n,k}(t) = 2\pi \sum_{\tau=0}^{t} f_{p,n,k}(\tau). \quad (8)$$

The inharmonic frequencies $\{f_{p,n,k}(t)\}_{k \leqslant K}$ are deduced from the fundamental frequency $f_{p,n,0}(t)$ and the inharmonicity coefficient $B_p(t)$ with the inharmonicity model of [35]:

$$f_{p,n,k}(t) = k f_{p,n,0}(t) \sqrt{1 + B_p(t) k^2}, \quad (9)$$

with the fundamental frequency $f_{p,n,0}$ obtained by converting the detuned pitch $\tilde{x}_p^{pitch} + \delta f_n$ with the MIDI note-to-frequency formula:

$$f_{p,n,0}(t) = 440 \times 2^{\frac{1}{12}(\tilde{x}_p^{pitch}(t) + \delta f_n(t) - 69)}. \quad (10)$$

The *subtractive synthesizer* generates the residual noises that happen during a performance, mainly the hammer and key noise upon note onsets, the pedal noises, and even the recording background noise. As in [11], a white noise $\boldsymbol{N}(t)$ is filtered in the frequency domain with the noise filter magnitudes $\boldsymbol{\eta}(t)$ computed by the model, before being inverted in the audio domain:

$$y_p^{noise}(t) = \text{DFT}^{-1}(\boldsymbol{\eta}_p(t) \boldsymbol{N}(t)). \quad (11)$$

The room response in the piano recordings is modeled by a differentiable convolutional reverberation. A finite impulse response IR_i is learned for each recording environment i, and it is applied to the sum of audio signals output by the bank of additive and subtractive synthesizers (Eq. (1)).

3 EXPERIMENTAL SETUP

3.1 Dataset

DDSP-Piano is trained and evaluated with performances from the MAESTRO dataset (v3.0.0) [1]. This dataset contains almost 200 h of professional piano performances spanning over $I = 10$ editions of the *International Piano-e-competition*. Pianists performed on Yamaha Disklaviers where MIDI data were recorded and aligned with the audio recordings. The ground-truth audio performances are downsampled to 16 kHz and downmixed to mono, which reduces the memory footprint of the training. Training at a higher frequency rate is possible and left for future works, at it requires to increase accordingly the number of harmonics and noise filter coefficients. The *una corda*, *sostenuto*, and *sustain* pedal controls are available in the MIDI recordings

through the 64, 66, and 67 *Control Change* (CC) messages, which corresponds to the three pedals of most grand pianos [26]. Conditioning and pedal controls are extracted with a frame rate of 250 Hz, as the conditioning controls in [11].

Tracks are split into 3-s-long segments, with a 50% overlap between two consecutive segments. Segments in which the maximum number of simultaneous notes is greater than the model polyphonic capacity P are removed from the training set.

3.2 Baseline Systems

The proposed DDSP-Piano model is evaluated against other piano sound synthesis methods presented in SEC. 1. All samples synthesized with the following systems are also downsampled to 16 kHz and converted to mono. For the sampling-based benchmark, performances are synthesized by stitching isolated note recordings from the `YDP Grand Piano`[2] soundfont, using the open-source software `Fluidsynth`.[3]

The commercial software `Pianoteq 7`[4] with the default preset `NY Steinway D Classical` is used as the physical-modeling-based baseline. Results from the physical modeling of the instrument are synthesized in real time using modal synthesis [28].

Finally, we chose the *Text-to-Speech* (TTS)-inspired model from [34] as the pure neural synthesizer benchmark. In particular, the `taco3-mfb-noi` variant has the best synthesis quality according to their evaluation. Also trained on the MAESTRO dataset, this model is a modified Tacotron-2 acoustic model [38] followed by a simplified *Neural Source Filter* (NSF) vocoder model [39]. MIDI-filter-bank–based spectra are used as the intermediate representation between the two submodels, which has the advantage of being aligned with the input piano rolls in the frequency/pitch axis.

3.3 Model Implementation Details

The proposed system is implemented with a polyphonic capacity of $P = 16$. The *Z-Encoder* outputs an embedding z_i of size 16 for each recording environment. The *context network* $\mathcal{F}$ is composed of a time-distributed dense layer of size 32 with leaky ReLU activation, followed by a causal *Gated Recurrent Unit* (GRU) layer of hidden size 64 and with layer normalization, then by a time-distributed linear layer outputting a context signal of size 32.

The *Note Release* module is initialized to extend the pitch conditioning control by $T_{release} = 1$s, which is longer than the observed attenuation time of piano notes after key release [40]. The *inharmonic model* is initialized with the parameters estimated in [35]: $\alpha_B^0 = -0.0847$, $\beta_B^0 = -5.82$, $\alpha_T^0 = 0.0926$ and $\beta_T^0 = -13.64$. We generate $n_{\text{strings}} = 2$ string signals per note, but their detuning are initialized to zero in the linear model g_δ of the *detuner*. The model-specific inharmonicity and detuning modifiers of the *Z-encoder* are also set first to zero, to learn a generic piano model during early training.

Table 1. Approximate number of trainable parameters of the evaluated neural-based models and their submodels.

Model	Parameters
Piano-TTS	31,335,000
- Tacotron-2	30,600,000
- NSF	736,300
DDSP-Piano	521,507
- Sub-models	281,507
- Reverb	240,000

The *monophonic model* $\mathcal{G}$ input is processed by a 128-unit time-distributed dense layer with leaky ReLU activation, then by a 192-unit GRU layer and another dense layer of size 192 with leaky ReLU activation. Layer normalization is then applied before computing the note amplitude, $K = 96$ harmonic amplitudes and 64 noise-filter coefficients with a linear layer.

Finally, the different reverb impulse responses are 1.5 s long at 16 kHz (24k parameters for each recording environment), with the same random initialization as in [14] and the inference decay function from [21]. The total number of training parameters in this *Default* configuration is given in Table 1, against Piano-*TTS*, the neural-based benchmark from SEC. 3.2.

3.4 Training

As with many recent neural-based audio synthesis methods [11, 14, 34], the model is trained to minimize the spectral difference between the target audio y and the synthesized audio $\hat{y}$ with a multiresolution spectral loss. The component $\mathcal{L}_m$ of the spectral loss with resolution m compares the two audio signals by summing the L1 differences between both their spectrogram magnitudes and log spectrograms, as in [11]:

$$\mathcal{L}_m(y, \hat{y}) = \| |\text{STFT}_m(y)| - |\text{STFT}_m(\hat{y})| \|_1 + \| \log |\text{STFT}_m(y)| - \log |\text{STFT}_m(\hat{y})| \|_1, \quad (12)$$

with $\|.\|_1$ the L1 norm and STFT_m the STFT with Fast Fourier Transform size $m \in \{2048, 1024, 512, 256, 128, 64\}$.

In preliminary experiments, it has been observed that the reverb module tried to model the notes' sustain and release behaviors, which resulted in abnormal reverberations and unrealistic unreverberated signals. A L1 regularization loss $\mathcal{L}_{L1}$ is applied on the learned impulse responses to reduce the reverb complexity and thus discouraging the module from learning characteristics of the piano tones (such note sustain and release) that can be modeled by the other unregularized modules.

Furthermore, the correct placement of partials in frequency is decisive for training stability, especially during early training. As partial frequencies in our system are deduced from explicit submodules, we propose a two-phase

[2]https://freepats.zenvoid.org/Piano/acoustic-grand-piano.html
[3]https://www.fluidsynth.org/
[4]https://www.modartt.com/pianoteq

training procedure for separately optimizing the pure *Deep Neural Network* (DNN) components and the explicit submodels.

During the first training phase, weights responsible for computing the partials frequencies are frozen to their initial values from SEC. 3.3, as they should be close to their optimal values. The concerned weights are those from the *detuner*, the *inharmonicity model*, and the model-specific detuning and inharmonicity modifiers of the *Z-encoder*. The other modules can thus learn to reproduce the notes spectral envelopes, residual noises, and the reverberation without displacing the note partials. The model is optimized using the Adam optimizer [41] with a learning rate of 10^{-3} and a batch size of 6, with regard to the first phase loss function $\mathcal{L}_1$:

$$\mathcal{L}_1 = \sum_m \mathcal{L}_m(y, \hat{y}) + \lambda_{\text{IR}} \mathcal{L}_{\text{L1}}(\text{IR}_i), \tag{13}$$

with λ_{IR} the balancing weight for the reverb regularization loss with regard to the spectral loss, here set to 0.01.

During the second training phase, the trainability of the model weights are reversed compared with the first training phase. In such a manner, the system should match the learned partials frequency and beating to each piano specifically. For training stability purpose, a L1 regularization loss is applied on the *inharmonicity model* parameters deviation from their initial values. The total loss associated with this second training phase can be expressed as:

$$\mathcal{L}_2 = \sum_m \mathcal{L}_m(y, \hat{y}) + \lambda_{\text{B}} \sum_{\theta \in \{\alpha_B, \beta_B, \alpha_T, \beta_T\}} |\theta - \theta^0|, \tag{14}$$

with $\lambda_B = 0.1$ the weight on the *inharmonicity model* regularization loss with regard to the spectral loss. The parameters of the *detuner*, *inharmonicity model*, and *Z-Encoder* model-specific modifiers are fine-tuned by Adam with a learning rate of 10^{-5} and a batching size of three.

The whole system is optimized and fine-tuned by successively alternating between these two training phases. In our experiments, the system is trained with the first phase formula for two full epochs on the 160 h of training data, until note partials are correctly generated by the additive synthesizer. It is then fine-tuned for one full epoch on the training data with the second training phase. Finally, the model is further fine-tuned using the first training phase again for three epochs, until the minimal validation loss value is reached. The full model training takes about 340k steps, which corresponds to around 8 days of training on a single Nvidia GeForce GTX 1080 Ti GPU.

3.5 Ablation Study

The relevance of our system submodules are evaluated by training and evaluating alternate versions of DDSP-Piano. All following variants are trained with the same procedure and hyper-parameters (losses balance, number of training steps, learning rates, batch sizes) exposed in SEC. 3.4.

The *Deep Inharmonicity* variant replaces the explicit *inharmonicity model* from [35] with by a DNN. Ideally, this DNN should reproduce Eq. (4): we use a *Multi-Layer Perceptron* (MLP) with sinusoidal activation functions as in [14]. This model takes the same inputs as the explicit *inharmonicity model*, and it is composed of three dense layers with sinusoidal activation and a hidden size of eight, followed by a linear layer with ReLU activation. The final output is scaled in order to keep the estimated inharmonicity factor within a realistic range $B \in [0, 0.02]$.

The *Reduced-context* variant imitates sampling-based synthesis by removing the conditioning input from the context vector computation. Because the synthesizer controls are computed on all monophonic channels independently, a monophonic note control would not have information on which other notes are also played, thus preventing mutual interaction between notes.

The *No Fine-tuning* variant is the *Default* configuration wherein the *inharmonicity model* and the *detuner* submodules are reverted back to their initial values before training. This variant is similar to a model trained solely with the first training phase of SEC. 3.4.

As stated in the original DDSP paper [11], the sound structure priors inherent to the DDSP synthesizers enable full model training with a reduced amount of data. In the same manner, we test DDSP-Piano on the simpler but also less resourceful task of single piano modeling. The *2009-only* variant is trained by only keeping performances from the year 2009 in the MAESTRO dataset, which amount for about 20 h of training data.

4 EVALUATION

Evaluations of the DDSP-Piano model has been conducted against its ablated variants from SEC. 3.5 and other piano synthesis methods from SEC. 3.2.

4.1 Objective Evaluation: Reconstruction Quality

The reconstruction quality of the model can be measured with the spectral difference between real and synthesized performances from the test subset. Using the multiresolution spectral loss presented in SEC. 3.4, the reconstruction quality of the model variants on each recording environment in the MAESTRO dataset are given in Fig. 2. An approximate per-note evaluation of the models is also presented in Fig. 3: for a given note, the approximate value is obtained by gathering the loss value of the test segments for each time the note is present before computing the mean.

On all recording environments and over the whole piano tessitura, the *Deep Inharmonicity* variant has higher loss values than variants using the explicit *inharmonicity model* from [35]. This explicit submodule proves to be valuable for the overall reconstruction quality of the model as it allows it to control the additive synthesizer with better estimated spectral envelopes.

The variant solely trained on performances from the year 2009 seems to have slightly better reconstruction quality on the 2009 subset compared with the *Default* configuration, although not significantly. If the difference is confirmed perceptually, that would suggests that the piano model embedding in the multi-instrument setting is not sufficient for

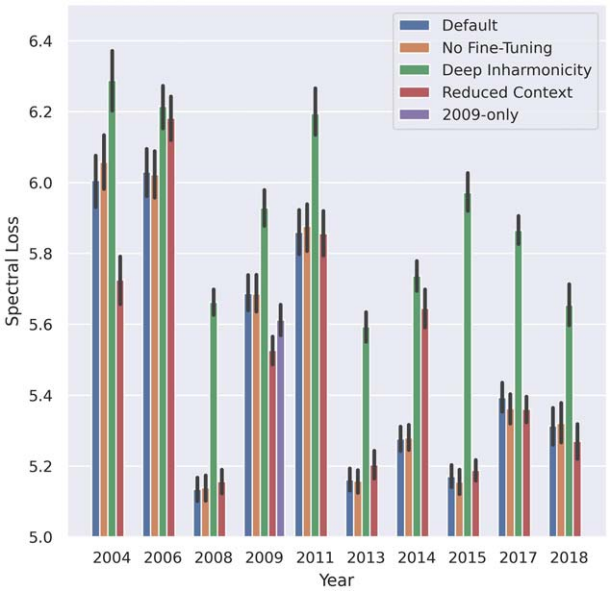

Fig. 2. Systems evaluation on the MAESTRO test set, broken down by recording environments. Measured by mean and standard deviation values of the multi-resolution spectral difference (as presented in SEC. 3.4) with the original recordings. Low loss value indicates better reconstruction quality.

achieving the same quality as for single-piano modeling. Nonetheless, if one wants to profile a single-piano model, training on the full MAESTRO dataset is not necessary, and smaller aligned datasets could be used instead [42].

Reducing the global context appears to have different consequences, as the *Reduced Context* variant can have similar, better, or worse reconstruction quality than the *Default* configuration depending on the piano model. Although mutual notes interaction should be present in all performances, independently of the recording year, the perceived effect may not be exhibited in the interpreted piano pieces. If modeling such interaction is unnecessary, the *Reduced Context* variant, with reduced complexity, can converge faster than the *Default* configuration and achieve better reconstruction quality for the same number of training epochs: this would be the case for pieces of the years 2004 and 2009. On the contrary, if such effects are significantly present in the training pieces, the ablated variant cannot reproduce it and thus achieves worse reconstruction quality, which concerns the years 2006 and 2014. Mutual interaction between notes (sympathetic resonances) is more exploited in contemporary music for instance: using such examples may help the system to systematically learn this specificity.

The approach performs similarly with and without applying the fine-tuned inharmonicity and detuning parameters. The loss profile over the piano tessitura is unchanged between the *Default* and *No Fine-Tuning* variant for all recordings years. This indicates that the second training phase proposed in SEC. 3.4 could not successfully fine-tune the inharmonicity and detuning parameters to the target pianos. Otherwise, the note partials would have been exactly matched in frequency, and the model could better reproduce the associated amplitude through the spectral loss, during the third training phase.

For all piano models, the loss curves along the tessitura show better reconstruction in the middle range than for lower- and higher-pitched notes. This behavior can come from the imbalance toward middle notes in the training data, as illustrated in Fig. 4, or also because of the imperfect estimation of the inharmonicity coefficients, which are higher in the lower- and higher-pitch ranges [35]. This indicates that the model could benefit from training data rebalancing, either by putting higher weights on the loss when training on underrepresented notes or by providing training segments with underrepresented notes more frequently. The highest notes (above pitch 100) can have slightly better reconstruction quality than the notes in the 80–100 pitch range, as the nonmatched inharmonic partials are located above the Nyquist frequency and thus ignored in the spectral loss computation.

In conclusion, the DDSP-Piano model can be trained for single-piano modeling and only with the first training phase

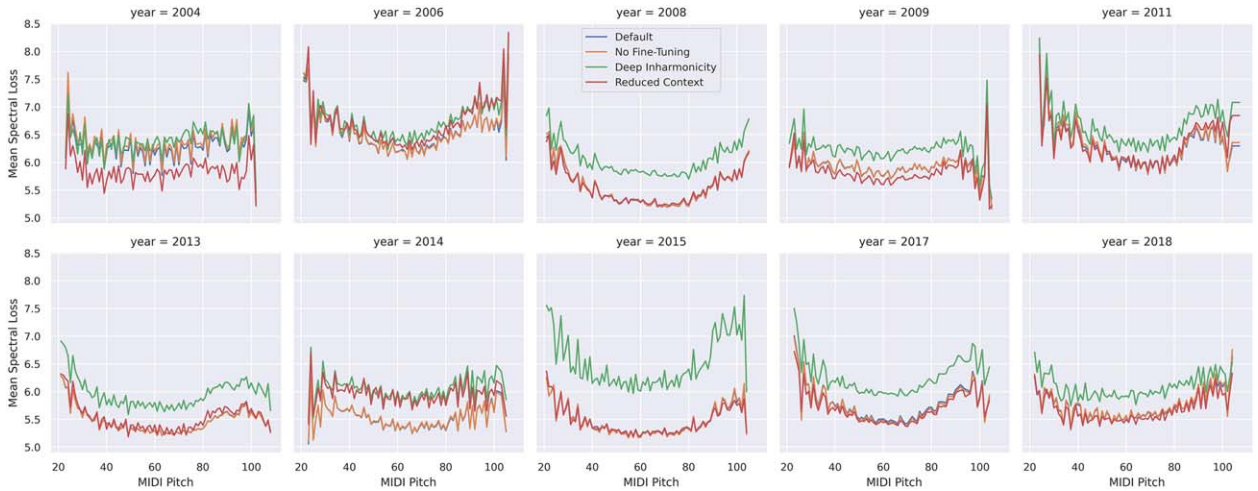

Fig. 3. Approximate mean values of note-wise spectral differences between real and synthesized audio samples. Results are detailed for each piano model in the test set. Low loss value indicates better reconstruction quality.

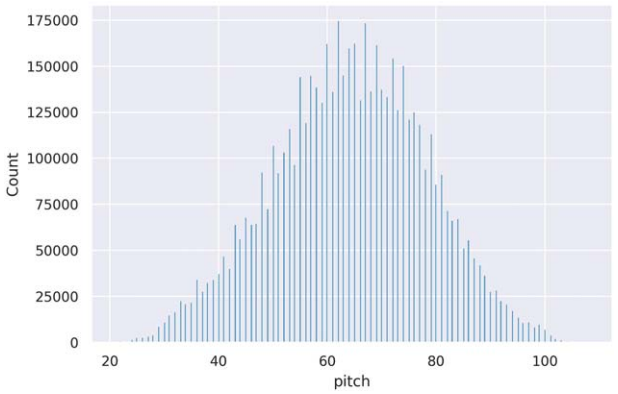

Fig. 4. Histogram of the notes distribution in the MAESTRO v3.0.0 training dataset.

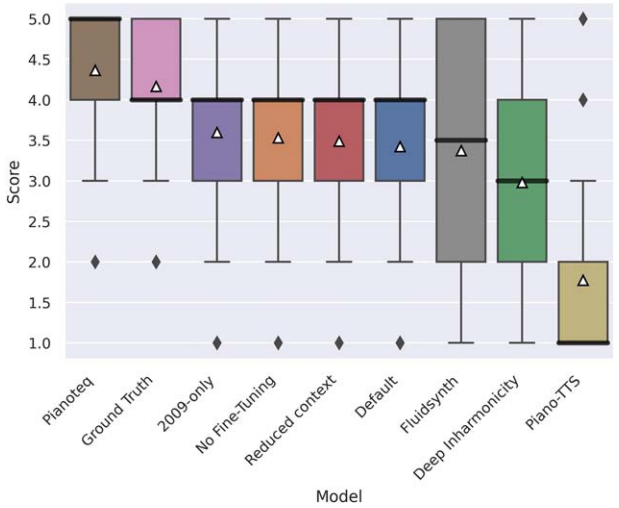

Fig. 5. Box-plots of MOS for each system. The thickened bars indicate the median values while the white triangles indicates the mean values.

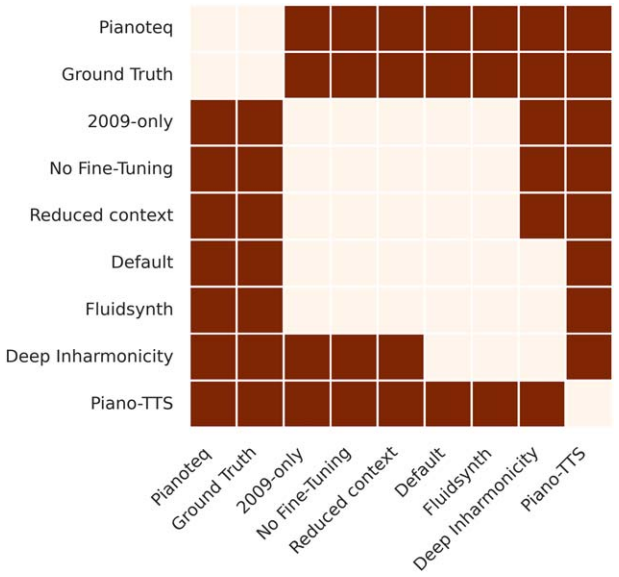

Fig. 6. Results of Holm–Bonferroni corrected two-sided Mann–Whitney U tests. A darkened block indicates a statistically significant difference at $\alpha = 0.05$ between two synthesis models.

from SEC. 3.4, without significant loss of reconstruction quality. Reducing the context may accelerate the training if mutual note interactions are ignored in the modeling. Nevertheless, the explicit *inharmonicity model* is crucial for reaching good reconstruction quality.

4.2 Subjective Evaluation: Listening Test

A listening test was conducted for gathering *Mean Opinion Score* (MOS) on all systems under evaluation. Eleven performances were hand-picked from the test data, covering all recording environments and with a diversity of composers, registers, and note densities. The first 9 s of the performances were synthesized with all systems, which, with the real recordings, gives 99 audio samples to evaluate. Listeners were asked to rate their overall quality with a scale from 1 (very annoying) to 5 (real recording). In each trial, two samples from each of the eight systems and two real recordings were randomly presented to the listener for rating. We gathered 52 participants who are musicians or audio professionals: 14 among them have notions of piano playing and 29 have been playing the instrument for several years. Box-plot and mean values of the MOS ratings are reported in Fig. 5, and Fig. 6 shows the results of two-sided Mann–Whitney U tests with Holm–Bonferroni correction, following the evaluation procedure of [34].

The quality difference between the *Deep Inharmonicity* variant and the models including the explicit *inharmonicity model* is confirmed perceptually. Only the *Default*-against-*Deep Inharmonicity* hypothesis is not statistically significant, but the median and quartile values still suggest a slight advantage in favor of the *Default* configuration. Ratings also confirm that the second training phase does not improve the perceived quality, suggesting that the natural beating between simultaneous notes in harmony may be sufficient for achieving realistic sounding partial beatings during polyphonic performances. Reducing the context also does not significantly hinder the perceived quality of the DDSP-Piano model. It can be deduced that other components of the approach can be improved before the lack of note interaction limits the perceived quality. Single-piano modeling is still perceived as good sounding as variants trained on several pianos simultaneously, which raises the question of the minimum amount of training data required for achieving such quality.

All variants of DDSP-Piano have a significant difference over the neural-based Piano-TTS benchmark. Although this baseline is more versatile because it was developed for speech synthesis at first, our approach is better suited for piano sound synthesis, achieving better sound quality with significantly less training parameters, as shown in Table 1.

Only the physical-modeling-based method achieves sound quality comparable with the real recordings (even slightly better, although not significantly, as in [34]). Various unwanted noises and the recording quality of the real samples may have been perceived as slightly annoying compared with the clean sounds synthesized by the `Pianoteq`

software. The quality of the training data represents the upper bound limit of neural-based synthesizers, thus our model can benefit from cleaner audio recordings. Nonetheless, there is still a significant gap in the perceived quality between the synthesis offered by the DDSP-Piano model and the real recordings. As it stands, all variants of our approach are not significantly different from the sampling-based synthesizer in terms of overall quality ratings, although with less variability. Among all evaluated systems, the ratings given to the synthesis from `Fluidsynth` are the most scattered: this may suggest that some listeners are more sensitive than others to an unrealistic feature in this synthesis algorithm.

4.3 Computational Load for Inference

The inference time of the *Default* model was measured on the test data to assess if the provided Tensorflow implementation[5] is capable of real-time synthesis. For all 3-s-long segments of the test data, the average synthesis time is measured on the same hardware as for GPU training in SEC. 3.3 (Nvidia GeForce GTX 1080 Ti) and on an 2.6 GHz Intel Xeon E5-2623 v4 CPU processor. We report real-time factors of 0.6 ± 0.1 on the GPU and 1.9 ± 0.1 on CPU. This invalidates the usage of the current implementation for real-time applications without relying on a GPU. However, model architecture design choices were made in order to reduce the structural latency of the model: the GRU layers do not rely on future samples as they are unidirectional, their hidden sizes are compatible for CPU computation in real-time [43], and the model does not rely on noncausal convolution operations as they raise a challenge for real-time applications [44]. Therefore, by exporting the learned model for inference into efficient frameworks dedicated to real-time neural audio processing [43, 45, 46], the real-time factor of DDSP-Piano on CPU can be improved.

5 QUALITATIVE RESULTS

As the synthesis relies on spectral modeling, it is possible to examine the behavior of the model and interpret what it has learned more easily than with pure deep learning models. To this end, the following section will put into relation different DDSP-synthesizers inputs and outputs with expected results from signal-based and acoustic modeling works.

5.1 Additive Synthesizer

For a single input note, the note amplitude envelope $a_p(t)$ predicted by the *default* configuration of DDSP-Piano is shown in Fig. 7. In the logarithmic scale, one can see that the amplitude decays at two distinct and successive rates: the piano note decays faster right after the onset before settling down to an aftersound with a slower decay. This corresponds to the "double decay" phenomenon [47], which is characteristic of piano notes. Previous piano modeling works have explained the "double decay" as a result of the double polarization of piano string and the detuning between substrings in a duet [37]. No prior knowledge of this effect was incorporated in the architecture of DDSP-Piano, but thanks to the recurrent layers, it has successfully captured it directly from the audio data. Thus, explicit modeling of the double decay is not necessary for improving the synthesis quality, but it can help reducing the number of training parameters nonetheless.

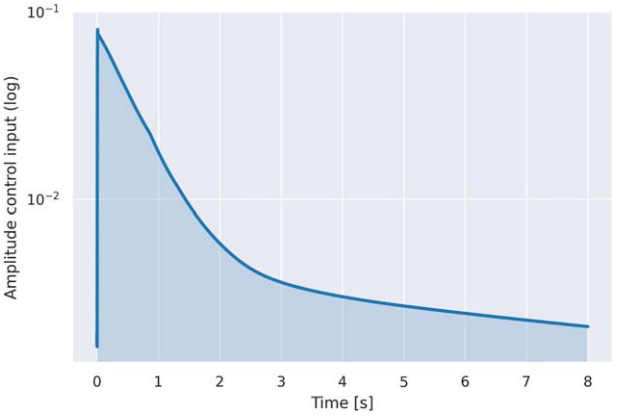

Fig. 7. Amplitude control input (in log scale) of the *additive synthesizer* predicted by the *default* model for a sustained A3 note with a velocity value of 100.

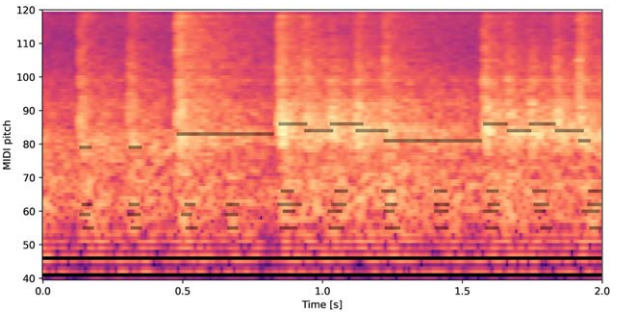

Fig. 8. MIDI-filter-bank–based spectra [34] of the *subtractive synthesizer* output. The active piano roll of the input performance is superposed over in black. Both are zoomed around the MIDI range [40,120].

5.2 Filtered Noise

The filtered noise output by the *subtractive synthesizer* from a real test performance input is shown in Fig. 8. The audio is represented with the MIDI-filter-bank-based spectra proposed in [34], which has the advantage of being aligned both in time and frequency with the input MIDI represented as a piano roll. Noise is synthesized in all frequency bands during note onset times, which corresponds to the impacts of the key on the keyboard base and the hammer on the string. However, the noise spectrum presents a correspondence between the played notes and the energy location in frequency, which can coincide with the piano soundboard modes. These modes have proven to be quite challenging to model because of the peculiar shape of the piano [48]. Indeed, the key and hammer impacts excite the

[5] https://github.com/lrenault/ddsp-piano

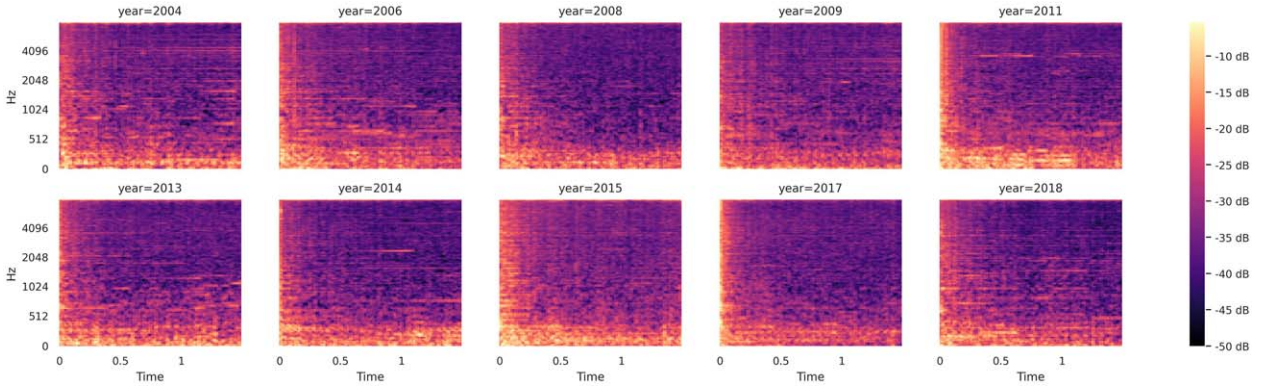

Fig. 9. Mel-spectrograms of the reverb impulse responses $\{IR_i\}_{i \leqslant I}$ learned by the *Default* configuration for each recording year in the MAESTRO dataset.

soundboard from different positions according to the note: higher modes are typically excited when higher notes are being played, as they are located in the shorter area of the soundboard curve. The model successfully replicated this relationship between the soundboard modes and the input pitch by learning from the target audio data, without the need for explicit modeling of the soundboard modes.

Several participants in the listening test have reported hearing a continuous background noise in the lower frequencies of the samples synthesized by the model. The authors have noticed, empirically, that this noise is output by the *subtractive synthesizer* and it is different from one piano model to another. This behavior may reflect the varying quality of the recordings in the MAESTRO dataset. One way to circumvent this issue would be to train the model with an auxiliary noise generator conditioned only on the recording environment label i. This constant noise generator would then be disabled during inference in order to obtain cleaner audio synthesis from the main model only.

5.3 Reverberation

Spectrograms of real room impulse responses have been analyzed and modeled as having high energy in all frequency bands during the early reflections and only in the lower frequency bands for the late reverberation [49]. As it can be seen in Fig. 9, some impulse responses learned by the reverb module of DDSP-Piano do not totally satisfy this modeling because one can distinguish modes in the higher frequencies in the late reverberation. When put into relation with the reconstruction quality presented in Fig. 2, the least-well-modeled pianos (most notably from the years 2006, 2004, and 2011) show such modes in their impulse responses estimated by the model. These modes may correspond to either prevalent note partials in the training data (as the authors observed before adding the L1 regularization constraint) or soundboard modes, which can also be simulated with reverberation algorithms [50].

In either case, such features should have been generated by other DDSP components instead. Despite the L1 regularization, the reverberation module remains too expressive and does not achieve a realistic sounding reverberation, as it also tries to model behaviors not related to the recording environment. Thus, using a differentiable reverberation model with added constraints, informed by room impulse response modeling works [18], can help achieving better behavior disentanglement between the DDSP components.

6 DISCUSSION

Training a neural piano synthesizer directly on polyphonic performance data enables the reproduction of complex interactions between different sound sources in the instrument. Previous works mainly focus on achieving realistic sounding synthesis by adapting state-of-the-art audio synthesis models. However, in order to better control the learned model, another challenging issue can be raised in the form of correctly disentangling the sound components, especially when the training data do not present these components separately.

In the continuity of DDSP [11], the proposed DDSP-Piano model further incorporates signal-based and acoustic-modeling knowledge into the differentiable framework for handling different instrument specificities. Such knowledge can be integrated as architectural constraints through explicit submodels (such as the inharmonicity model presented in SEC. 2.3), layers connections motivated by meaningful inputs and outputs (by knowing which variable contributes to which phenomenon) and variable processing (with the *monophonic network* being applied on each monophonic channel for example). The model successfully achieves better sounding quality than a pure neural-based synthesizer and shows promising results for disentangling the different sound components.

After conducting evaluations against ablated variants and observing qualitative results from the interpretable DDSP components, the need for improving the optimization process has been revealed, in order for the submodules to correctly match their expected behaviors. In particular, training through minimization of the spectral loss is not sufficient for pitch estimation [51]: this hinders the optimization of the inharmonicity and detuning submodules in an end-to-end manner. On the other hand, very expressive submodules can help the synthesizer reproduce behaviors that are difficult to model explicitly. Yet they can also be responsible for the

imperfect disentanglement of sound components between each other. Such behavior could be regularized by adding constraints, through architectural design choices with more specialized or explicit submodules or through optimization constraints with informed losses.

Finally, the integrated constraints require the data to fulfil associated assumptions: here, DDSP-Piano would not be able to profile instruments with extreme properties, such as highly detuned pianos. On the other hand, if the data fulfill the assumptions, less training data will be required when compared with the case when models do not incorporate any constraints. Future physics-informed models should thus balance the neural networks' expressivity and the instrument knowledge constraints to the quantity of training data available and the desired model flexibility.

7 CONCLUSION

This work introduces an extension of DDSP models for the task of polyphonic instrument modeling. A realistic-sounding piano synthesizer is conceived using high-level modeling knowledge in order to combine expressive neural networks with explicit submodules that take care of piano specificities, such as the inharmonicity of note partials. In a subjective evaluation, this hybrid model, with significantly less parameters, has achieved better synthesis quality than a state-of-the-art neural model but does not outperform sampling-based and physical-modeling-based systems.

Thanks to its interpretability, further analysis of the model behavior has been conducted, which has revealed that some results found in acoustic modeling works were reproduced by the model. It has also been revealed that the different sound components were not as well disentangled as the model architecture would allow, which opens up perspectives for further integration of acoustic modeling knowledge as constraints and for adapting the training procedure with regard to such constraints. Future works could also leverage the interpretability and the differentiability of DDSP-Piano to address other polyphonic music-related tasks, such as source separation [23] and self-supervised multi-pitch transcription [52].

8 ACKNOWLEDGMENTS

This work was supported by European Union's Horizon 2020 research and innovation program under grant number 951911 - AI4Media. Part of this work was performed using High-performance computing resources from GENCI-IDRIS (Grant 2022-AD011013202). We would like to thank Erica Cooper, among the authors from [34], for kindly sharing their test samples for the listening test and thank to the Conference on Digital Audio Effects (DAFx20in22) committee for providing the open access publication of this work.

9 REFERENCES

[1] C. Hawthorne, A. Stasyuk, A. Roberts, et al., "Enabling Factorized Piano Music Modeling and Generation with the MAESTRO Dataset," in *Proceedings of the International Conference on Learning Representations (ICLR)* (New Orleans, LA) (2019 May).

[2] A. Défossez, N. Zeghidour, N. Usunier, L. Bottou, and F. Bach, "SING: Symbol-to-Instrument Neural Generator," in *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*, pp. 9055–9065 (Montréal, Canada) (2018 Dec.).

[3] J. Engel, C. Resnick, A. Roberts, et al., "Neural Audio Synthesis of Musical Notes With WaveNet Autoencoders," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pp. 1068–1077 (Sydney, Australia) (2017 Aug.).

[4] P. Esling, A. Chemla-Romeu-Santos, and A. Bitton, "Bridging Audio Analysis, Perception and Synthesis with Perceptually-regularized Variational Timbre Spaces," in *Proceedings of the 19th International Society for Music Information Retrieval (ISMIR)*, pp. 175–181 (Paris, France) (2018 Sep.).

[5] J. Engel, K. K. Agrawal, S. Chen, et al., "GANSynth: Adversarial Neural Audio Synthesis," in *Proceedings of the Seventh International Conference on Learning Representations (ICLR)* (New Orleans, LA) (2019 May).

[6] J. Nistal, S. Lattner, and G. Richard, "DrumGAN: Synthesis of Drum Sounds With Timbral Feature Conditioning Using Generative Adversarial Networks," in *Proceedings of the 21st International Society for Music Information Retrieval (ISMIR)*, pp. 590–597 (Montréal, Canada) (2020 Oct.).

[7] A. Lavault, A. Roebel, and M. Voiry, "StyleWaveGAN: Style-based Synthesis of Drum Sounds with Extensive Controls using Generative Adversarial Networks," in *Proceedings of the Sound and Music Computing Conference (SMC)*, pp. 365–372 (Saint-Étienne, France) (2022 Jun.). https://doi.org/10.5281/zenodo.6798171.

[8] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, "Diffwave: A Versatile Diffusion Model for Audio Synthesis," in *Proceedings of the International Conference on Learning Representations (ICLR)* (Vienna, Austria) (2021 May).

[9] J. Liu, C. Li, Y. Ren, F. Chen, P. Liu, and Z. Zhao, "Diffsinger: Singing Voice Synthesis via Shallow Diffusion Mechanism," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 10, pp. 11020–11028 (2022 Jun.). https://doi.org/10.1609/aaai.v36i10.21350.

[10] C. Hawthorne, I. Simon, A. Roberts, et al., "Multi-Instrument Music Synthesis with Spectrogram Diffusion," in *Proceedings of the 23rd International Society of Music Information Retrieval (ISMIR)*, pp. 598–607 (Bengaluru, India) (2022 Dec.).

[11] J. Engel, L. H. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable Digital Signal Processing," in *Proceedings of the International Conference on Learning Representations (ICLR)* (Virtual/Addis Abada, Ethiopia) (2020 Apr./May).

[12] X. Serra and J. O. Smith, "Spectral Modeling Synthesis: A Sound Analysis/Synthesis System Based on a Deterministic Plus Stochastic Decomposition," *Comput. Music J.*, vol. 14, no. 4, pp. 12–24 (1990 Winter).

[13] S. Shan, L. Hantrakul, J. Chen, M. Avent, and D. Trevelyan, "Differentiable Wavetable Synthesis," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4598–4602 (Singapore, Singapore) (2022 May). https://doi.org/10.1109/ICASSP43922.2022.9746940.

[14] B. Hayes, C. Saitis, and G. Fazekas, "Neural Waveshaping Synthesis," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, pp. 254–261 (Online) (2021 Nov.).

[15] F. Caspe, A. McPherson, and M. Sandler, "DDX7: Differentiable FM Synthesis of Musical Instrument Sounds," in *Proceedings of the 23rd International Society for Music Information Retrieval Conference (ISMIR)*, pp. 608–616 (Bengaluru, India) (2022 Dec.).

[16] B. Kuznetsov, J. D. Parker, and F. Esqueda, "Differentiable IIR Filters for Machine Learning Applications," in *Proceedings of the 23rd International Conference of Digital Audio Effects (DAFx)*, pp. 297–303 (Vienna, Austria) (2020 Sep.).

[17] S. Nercessian, "Neural Parametric Equalizer Matching using Differentiable Biquads," in *Proceedings of the 23rd International Conference on Digital Audio Effects (DAFx)*, pp. 265–272 (Vienna, Austria) (2020 Sep.).

[18] S. Lee, H.-S. Choi, and K. Lee, "Differentiable Artificial Reverberation," *IEEE Trans. Audio Speech Lang. Process.*, vol. 30, pp. 2541–2556 (2022 Jul.). https://doi.org/10.1109/TASLP.2022.3193298.

[19] R. Castellon, C. Donahue, and P. Liang, "Towards Realistic MIDI Instrument Synthesizers," in *Proceedings of the Fourth NeurIPS Workshop on Machine Learning for Creativity and Design* (Vancouver, Canada) (2020 Dec.).

[20] N. Jonason, B. Sturm, and C. Thomé, "The Control-Synthesis Approach for making Expressive and Controllable Neural Music Synthesizers," in *Proceedings of the First Joint Conference on AI Music Creativity* (Stockholm, Sweden) (2020 Oct.). https://doi.org/10.5281/zenodo.4285384.

[21] Y. Wu, E. Manilow, Y. Deng, et al., "MIDI-DDSP: Hierarchical Modeling of Music for Detailed Control," in *Proceedings of the Tenth International Conference on Learning Representations (ICLR)* (Online) (2022 Apr.).

[22] M. Kawamura, T. Nakamura, D. Kitamura, et al., "Differentiable Digital Signal Processing Mixture Model for Synthesis Parameter Extraction from Mixture of Harmonic Sounds," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 941–945 (Singapore, Singapore) (2022 May). https://doi.org/10.1109/ICASSP43922.2022.9746399.

[23] K. Schulze-Forster, C. S. Doire, G. Richard, and R. Badeau, "Unsupervised Audio Source Separation Using Differentiable Parametric Source Models," *IEEE Trans. Audio Speech Lang. Process.*, vol. 31, pp. 1276–1289 (2023 Mar.). https://doi.org/10.1109/TASLP.2023.3252272.

[24] L. Renault, R. Mignot, and A. Roebel, "Differentiable Piano Model for MIDI-to-Audio Performance Synthesis," in *Proceedings of the 25th International Conference on Digital Audio Effects (DAFx)*, pp. 232–239 (Vienna, Austria) (2022 Sep.).

[25] D. Schwarz, "Concatenative Sound Synthesis: The Early Years," *J. New Music Res.*, vol. 35, no. 1, pp. 3–22 (2006 Mar.). https://doi.org/10.1080/09298210600696857.

[26] J. Chabassier, *Modélisation et Simulation Numérique d'un Piano par Modèles Physiques*, Ph.D. thesis, École Polytechnique X, Palaiseau, France (2012 Mar.).

[27] J. Rauhala, M. Laurson, V. Välimäki, H.-M. Lehtonen, and V. Norilo, "A Parametric Piano Synthesizer," *Comput. Music J.*, vol. 32, no. 4, pp. 17–30 (2008 Winter).

[28] B. Bank and J. Chabassier, "Model-Based Digital Pianos: From Physics to Sound Synthesis," *IEEE Signal Processing Mag.*, vol. 36, no. 1, pp. 103–114 (2019 Jan.). https://doi.org/10.1109/MSP.2018.2872349.

[29] J. O. Smith and X. Serra, "PARSHL: An Analysis/Synthesis Program for Non-Harmonic Sounds Based on a Sinusoidal Representation," in *Proceedings of the 13th International Computer Music Conference (ICMC)* (Champaign/Urbana, IL) (1987 Aug.).

[30] H. Hahn and A. Roebel, "Extended Source-Filter Model for Harmonic Instruments for Expressive Control of Sound Synthesis and Transformation," in *Proceedings of the 16th International Conference on Digital Audio Effects (DAFx)* (Maynooth, Ireland) (2013 Sep.).

[31] J. W. Kim, R. Bittner, A. Kumar, and J. P. Bello, "Neural Music Synthesis for Flexible Timbre Control," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 176–180 (Brighton, UK) (2019 May). https://doi.org/10.1109/ICASSP.2019.8683596.

[32] H.-W. Dong, C. Zhou, T. Berg-Kirkpatrick, and J. McAuley, "Deep Performer: Score-to-Audio Music Performance Synthesis," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 951–955 (Singapore, Singapore) (2022 May). https://doi.org/10.1109/ICASSP43922.2022.9747217.

[33] Y.-J. L. Hao Hao Tan and D. Herremans, "Generative Modelling for Controllable Audio Synthesis of Expressive Piano Performance," in *Proceedings of the ICML Workshop on Machine Learning for Media Discovery Workshop (ML4MD)* (Vienna, Austria) (2020 Jul.).

[34] E. Cooper, X. Wang, and J. Yamagishi, "Text-to-Speech Synthesis Techniques for MIDI-to-Audio Synthesis," in *Proceedings of the 11th ISCA Speech Synthesis Workshop (SSW 11)*, pp. 130–135 (Budapest, Hungary) (2021 Aug). https://doi.org/10.21437/SSW.2021-23.

[35] F. Rigaud, B. David, and L. Daudet, "A Parametric Model of Piano Tuning," in *Proceedings of the 14th International Conference on Digital Audio Effects (DAFx)*, pp. 393–399 (Paris, France) (2011 Sep.).

[36] R. W. Young, "Inharmonicity of Plain Wire Piano Strings," *J. Acoust. Soc. Am.*, vol. 24, no. 3, pp. 267–273 (1952 May). https://doi.org/10.1121/1.1906888.

[37] G. Weinreich, "Coupled Piano Strings," *J. Acoust. Soc. Am.*, vol. 62, no. 6, pp. 1474–1484 (1977 Dec.). https://doi.org/10.1121/1.381677.

[38] J. Shen, R. Pang, R. J. Weiss, et al., "Natural TTS Synthesis by Conditioning Wavenet on MEL Spectrogram Predictions," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4779–4783 (Calgary, Canada) (2018 Apr.). https://doi.org/10.1109/ICASSP.2018.8461368.

[39] X. Wang, S. Takaki, and J. Yamagishi, "Neural Source-Filter-Based Waveform Model for Statistical Parametric Speech Synthesis," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5916–5920 (Brighton, UK) (2019 May). https://doi.org/10.1109/ICASSP.2019.8682298.

[40] H.-M. Lehtonen, A. Askenfelt, and V. Välimäki, "Analysis of the Part-Pedaling Effect in the Piano," *J. Acoust. Soc. Am.*, vol. 126, no. 2, pp. EL49–EL54 (2009 Aug.). https://doi.org/10.1121/1.3162438.

[41] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proceedings of the Third International Conference for Learning Representations (ICLR)* (San Diego, CA) (2015 May).

[42] V. Emiya, N. Bertin, B. David, and R. Badeau, "MAPS - A Piano Database for Multipitch Estimation and Automatic Transcription of Music," Research Report, INRIA (2010 Jul.). https://hal.inria.fr/inria-00544155.

[43] A. Wright, E.-P. Damskägg, and V. Välimäki, "Real-Time Black-Box Modelling With Recurrent Neural Networks," in *Proceedings of the 22nd International Conference on Digital Audio Effects (DAFx)* (Birmingham, UK) (2019 Sep.).

[44] A. Caillon and P. Esling, "Streamable Neural Audio Synthesis with Non-Causal Convolutions," in *Proceedings of the 25th International Conference on Digital Audio Effects (DAFx)*, pp. 320–327 (Vienna, Austria) (2022 Sep.).

[45] J. Chowdhury, "RTNeural: Fast Neural Inferencing for Real-Time Systems," *arXiv preprint arXiv:2106.03037* (2021).

[46] D. Stefani, S. Peroni, and L. Turchet, "A Comparison of Deep Learning Inference Engines for Embedded Real-time Audio Classification," in *Proceedings of the 25th International Conference on Digital Audio Effects (DAFx)*, pp. 256–263 (Vienna, Austria) (2022 Sep.).

[47] D. W. Martin, "Decay Rates of Piano Tones," *J. Acoust. Soc. Am.*, vol. 19, no. 4, pp. 535–541 (1947 Jul.). https://doi.org/10.1121/1.1916547.

[48] J. Chabassier, A. Chaigne, and P. Joly, "Modeling and Simulation of a Grand Piano," *J. Acoust. Soc. Am.*, vol. 134, no. 1, pp. 648–665 (2013 Jul.). https://doi.org/10.1121/1.4809649.

[49] V. Välimäki, J. D. Parker, L. Savioja, J. O. Smith, and J. S. Abel, "Fifty Years of Artificial Reverberation," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 20, no. 5, pp. 1421–1448 (2012 Jul.). https://doi.org/10.1109/TASL.2012.2189567.

[50] B. Bank, S. Zambon, and F. Fontana, "A Modal-Based Real-Time Piano Synthesizer," *IEEE Trans. Audio Speech Lang. Process.*, vol. 18, no. 4, pp. 809–821 (2010 May). https://doi.org/10.1109/TASL.2010.2040524.

[51] J. Turian and M. Henry, "I'm Sorry for Your Loss: Spectrally-Based Audio Distances Are Bad at Pitch," in *Proceedings of the First "I Can't Believe It's Not Better!" (ICBINB) Workshop at the International Conference on Neural Information Processing Systems (NeurIPS)* (Vancouver, Canada) (2020 Dec.).

[52] J. Engel, R. Swavely, L. H. Hantrakul, A. Roberts, and C. Hawthorne, "Self-Supervised Pitch Detection by Inverse Audio Synthesis," presented at the *ICML Workshop on Self-supervision in Audio and Speech* (Online) (2020 Jul.).

THE AUTHORS


Lenny Renault


Rémi Mignot


Axel Roebel

Lenny Renault is a doctoral student of Sorbonne University, in the Analysis/Synthesis team at IRCAM (UMR 9912 - STMS), Paris, France. His research interests include neural audio synthesis, symbolic music processing, and music information retrieval. He has worked previously as a research scientist intern at Deezer. In 2020, he received his Diplôme d'Ingénieur from Télécom Paris, as well as an M.Sc. in Informatics from Sorbonne University.

•

Rémi Mignot is a tenured researcher at IRCAM (UMR 9912 - STMS) in Paris, France, and a member of the Analysis/Synthesis team. His research expertise focuses on machine learning and signal processing applied to audio processing and indexing. He received a Ph.D. in Signal and Image Processing of Télécom ParisTech with IRCAM in 2009. Then, he did his first post-doctoral research in the Langevin Institut (ESPCI ParisTech and UPMC in Paris), where he studied the sampling of rooms impulses responses using Compressed Sensing, and a second post-doctoral research at Aalto University in Espoo, Finland, with a Marie Curie post-doctoral fellowship, where he worked on the sound synthesis of musical instruments based on an "Extended" Subtractive Synthesis approach. In 2014, he came back at IRCAM to work on audio indexing and music information retrieval, and he has obtained a permanent position since 2018.

•

Axel Roebel is Research Director at IRCAM and head of the Analysis/Synthesis team. He received his Ph.D. degree in 1993 from Technical University of Berlin, and his habilitation from University Pierre and Marie Curie (UPMC) in Paris in 2013. He has developed state-of-the-art speech and music analysis and transformation algorithms and has strong research interests in DNN based voice analysis, vocoders, singing synthesis, and music transcription. He has directed IRCAM's research in multiple European, ANR and industrial projects, as, for example, the FP7 ICT 2011 project 3DTVS, and the project ChaNTeR (ANR-13-CORD-0011).