BIQUAD COEFFICIENTS OPTIMIZATION VIA KOLMOGOROV-ARNOLD NETWORKS

Ayoub Malek
Automotive laboratory
Huawei Munich Research Center
Munich, DE
ayoub.malek@huawei.com

Donald Schulz
Automotive laboratory
Huawei Munich Research Center
Munich, DE
donald.schulz@huawei.com

Felix Wuebbelmann
Automotive laboratory
Huawei Munich Research Center
Munich, DE
felix.wuebbelmann@huawei.com

ABSTRACT

Conventional Deep Learning (DL) approaches to Infinite Impulse Response (IIR) filter coefficients estimation from arbitrary frequency response are quite limited. They often suffer from inefficiencies such as tight training requirements, high complexity, and limited accuracy. As an alternative, in this paper, we explore the use of Kolmogorov-Arnold Networks (KANs) to predict the IIR filter—specifically biquad coefficients—effectively. By leveraging the high interpretability and accuracy of KANs, we achieve smooth coefficients' optimization. Furthermore, by constraining the search space and exploring different loss functions, we demonstrate improved performance in speed and accuracy. Our approach is evaluated against other existing differentiable IIR filter solutions. The results show significant advantages of KANs over existing methods, offering steadier convergences and more accurate results. This offers new possibilities for integrating digital infinite impulse response (IIR) filters into deep-learning frameworks.

1. INTRODUCTION

IIR filters are widely used in control systems, audio processing, and communications due to their efficiency, low memory usage, and ability to emulate analog filters. They achieve sharp frequency responses with fewer coefficients than FIR filters, making them ideal for real-time tasks (despite stability and phase limitations) such as equalization, tone matching, and feedback reduction, etc. [1, 2]. However, existing methods for designing and estimating IIR filter coefficients from an arbitrary frequency response typically rely on heuristic or lengthy and slow iterative processes, which can be computationally expensive. These methods are particularly challenging in scenarios that require high precision and fast processing times [3]. Existing IIR filter design approaches usually focus on specific filter prototypes (e.g. low-pass filters, all poles filters, etc.) and employ techniques like modified Yule-Walker (MYW) estimation [4], least squares [5], linear programming, and gradient-based optimization [6, 7]. Although these methods can achieve reasonable accuracy, they often have inherent limitations. For example, the MYW method is computationally efficient, but may yield inaccurate results when the target response exhibits a complex structure [8]. On the other hand, iterative methods tend to offer higher accuracy but are computationally expensive and susceptible to challenges posed by non-convex optimization landscapes, leading to sensitivity to initial conditions and the risk of getting stuck in local minima [9].

Recent advances in signal processing and machine learning have led to the development of methods that integrate traditional Digital Signal Processing (DSP) techniques within deep learning frameworks. Among these innovations, the inclusion of IIR filters —particularly biquad filters— into Differentiable DSP (DDSP) models [10]. This promising combination addresses these limitations by incorporating differentiable operations into filter design. This enables the direct estimation of IIR filter coefficients within a neural network framework, allowing for backpropagation-based optimization [11]. Engel et al. [10] demonstrated that DDSP components, such as FIR filters and oscillators, can be seamlessly integrated into deep learning models, significantly enhancing both training efficiency and model performance. This integration leverages differentiable operations that enable powerful gradient-based optimization techniques while maintaining computational efficiency and supporting innovative neural network architectures. Notably, Recurrent Neural Networks (RNNs) have emerged as particularly effective tools for IIR filter design [12]. Taking advantage of their structural similarities to IIR filters, RNNs can learn arbitrary filter behaviors directly from data, effectively capturing the recursive dynamics of IIR filters [13]. This capability establishes RNNs as a robust framework for designing flexible and computationally efficient filtering systems [13, 14]. However, these models often encounter difficulties due to the recursive operations inherent to RNNs, requiring multiple gradient updates over time. Due to these recursive operations, RNNs suffer from various gradient-related issues (e.g., vanishing gradient, exploding gradient), making the training process slow and complex [15, 16]. Recent research has sought to mitigate these challenges by training neural networks to directly estimate the parameters of graphic and parametric equalizers. This involves training models to estimate the coefficients of IIR filters from their frequency responses [14]. While these approaches avoid some computational bottlenecks, they may still be limited by the specific estimated IIR filter prototype (e.g. all-poles, shelving, peak filter, etc.). [17, 18, 19], or they lack flexibility and accuracy due to their dependency on training data [8].

The promising results reported in [8, 13] highlight the potential of deep learning-based approaches for filter design. Building on these foundations, we propose a novel KAN-based framework for biquad IIR filter design that directly estimates filter coefficients through differentiable filtering. To ensure stability and improve performance, we introduce constrained search spaces and explore a range of loss functions, achieving enhancements in both speed and accuracy. We conduct a comprehensive evaluation against state-of-the-art differentiable IIR design methods, including RNN-based [13] and MLP-based [8]

approaches, across a diverse benchmark of 10 test filters. Our results demonstrate that the KAN-based method achieves faster convergence and lower approximation error. Additionally, we provide an interpretable analysis of pole-zero distributions and the effects of key hyperparameters.

2. BACKGROUND

2.1. Biquad Coefficients (BA) Representation

The transfer function of the biquad filter can be represented via its BA coefficients and substituting $z = e^{-j2\pi f}$ as follows:

$$H(f) = \frac{b_0 + b_1 e^{-j2\pi f} + b_2 e^{-j4\pi f}}{1 + a_1 e^{-j2\pi f} + a_2 e^{-j4\pi f}}, \tag{1}$$

with f the frequency in Hz and $e^{-j2\pi f}$ represents the complex exponential corresponding to the frequency f. As for the magnitude and phase responses, they are then calculated as follows:

$$|H(f)| = \left| \frac{b_0 + b_1 e^{-j2\pi f} + b_2 e^{-j4\pi f}}{1 + a_1 e^{-j2\pi f} + a_2 e^{-j4\pi f}} \right| \tag{2}$$

$$\angle H(f) = \text{atan2}\left(\text{Im}(H(f)),\, \text{Re}(H(f))\right) \tag{3}$$

where $\text{atan2}(y, x)$ is the four-quadrant inverse tangent, which computes the phase of the complex response $H(f)$.

2.2. Zero-Pole-Gain (ZPK) Representation

A biquad filter can also be described using its **zeros** and **poles**, which represent the roots of the numerator and denominator polynomials, respectively. The transfer function in terms of zeros and poles is given by:

$$H(z) = G \cdot \frac{(z - z_1)(z - z_2)}{(z - p_1)(z - p_2)}, \tag{4}$$

where, G is the system gain, z_1 and z_2 are the zeros of the filter, and p_1 and p_2 are the poles of the filter. The zeros and poles are typically represented as complex numbers. Hence, each zero z and pole p can be expressed as:

$$z = q_0 + jq_1, \quad p = p_0 + jp_1$$

with q_0, q_1 being the real and imaginary parts of the zero and p_0, p_1 are the real and imaginary parts of the pole, respectively. Here, j denotes the imaginary unit.

2.3. Conversion of Zeros and Poles to Biquad Coefficients

The feedforward coefficients b_0, b_1, b_2 are computed from the zero positions as follows:

$$\begin{aligned} b_0 &= G \\ b_1 &= -2 \cdot G \cdot \text{Re}(z) \\ b_2 &= G \cdot \left[\text{Re}(z)^2 + \text{Im}(z)^2\right] \end{aligned} \tag{5}$$

where G is the system gain and $\text{Re}(z)$ and $\text{Im}(z)$ are the real and imaginary parts of the zero z, respectively. Similarly, the feedback coefficients a_1, a_2 are computed from the pole positions as:

$$\begin{aligned} a_1 &= -2 \cdot \text{Re}(p) \\ a_2 &= \text{Re}(p)^2 + \text{Im}(p)^2 \end{aligned} \tag{6}$$

where $\text{Re}(p)$, $\text{Im}(p)$ are the real and imaginary parts of the pole p. The conversion between the ZPK and BA forms involves either combining the zeros and poles to form the biquad transfer function (ZPK to BA) or solving the quadratic equations for each second-order section (BA to ZPK). This allows flexibility in filter implementation, as each form has its own advantages depending on the context, such as ease of implementation, numerical stability, or filter design objectives.

2.4. Neural Network Approach for Filter Design

The application of neural networks for filter coefficient estimation often involves a differentiable reimplementation of DSP, which can be integrated with traditional neural networks such as Multilayer Perceptrons (MLPs) [8]. While MLPs have proven to be a robust and versatile choice for various problems, the level of interpretability required in this context, combined with the non-convex nature of the problem, suggests that MLPs might not be the optimal solution. To address these limitations, we introduce Kolmogorov Arnold networks (KANs), a novel paradigm of neural networks where the weights are fixed and the activations are trained. KANs have shown superior performance in symbolic formula representation and offer significant interpretability advantages over traditional neural networks [20, 21].

3. EXISTING APPROACHES

Conventional methods for computing biquad filter coefficients typically rely on numerical optimization techniques or closed-form solutions derived from specific filter design specifications. Although effective for static filtering tasks, these methods face limitations when integrated into adaptive systems, such as neural networks, which require flexibility and real-time learning capabilities. Recent research has shifted towards differentiable filter design via Differentiable Digital Signal Processing. DDSP combines traditional Digital Signal Processing (DSP) with Deep Learning (DL), offering novel approaches to estimating filter parameters in dynamic and adaptive settings. This paradigm reimplements traditional DSP algorithms using differentiable operations, enabling the integration of classic DSP techniques, such as filtering and oscillation, into deep learning models. Through this framework, backpropagation can be used for optimization [10]. DDSP has been applied to various signal processing tasks, including the estimation of filter coefficients, where it improves the direct learning of filter parameters within a neural network architecture. A significant body of research has focused on using DDSP for filter parameter estimation. Kuznetsov et al. demonstrated how Recurrent Neural Networks (RNNs) can provide a practical implementation of differentiable IIR filters [13]. Most other studies in this domain have targeted specific filter types, such as all-pass and all-pole filters. For example, Bargum et al. [18] and Yu et al. [19] explored differentiable models to estimate the coefficients of these specialized filters, showcasing their potential for adaptive signal processing. These models optimize filter coefficients using gradient-based methods, offering a more efficient and scalable solution compared to traditional optimization-based approaches. The research by Mockenhaupt et al. [22] and Nercessian et al. [9] investigated neural network-based approaches combined with DDSP elements for automatic equalization in various audio processing scenarios. These studies predict biquad coefficients using neural networks, highlighting the ability of DDSP to learn

filter parameters directly from data. This is particularly beneficial in dynamic environments where traditional methods struggle to adapt. Another key area of research emphasizes the flexibility of DDSP-based methods in filter design. Colonel et al. [8] argued that filter design should not be restricted to specific filter prototypes, as is often the case with traditional methods. Instead, they proposed a direct design approach using Multilayer Perceptrons (MLPs) to estimate biquad filter coefficients. This method adapts to a wide range of filter types and responses, enabling the development of more general-purpose models applicable to various signal processing tasks without being constrained by predefined filter structures. In conclusion, this shift to DDSP opens up new possibilities for accurate, data-driven filter coefficient estimation from arbitrary frequency responses; therefore, surpassing the limitations of classic optimization techniques. However, none of these approaches leverage the symbolic interpretability of KANs; our method fills this gap by using KANs to directly estimate filter parameters, combining the flexibility of DDSP with KAN's structured representation.

4. PROPOSED APPROACH

Inspired by the approaches of Kuznetsov et al. [13], Colonel et al. [8] and insights from prior research, we introduce a hybrid optimization method that leverages Kolmogorov–Arnold Networks (KANs) [20]. Known for their interpretability, KANs provide significant advantages in domains where transparency and adaptability are critical [21]. Our method combines the high precision of iterative optimization with the speed and predictive power of neural networks. The central idea of our approach is the direct estimation of IIR filter coefficients from its frequency response. Starting with the frequency response, we simulate the filtering of an input signal and iteratively adjust the filter coefficients based on the resulting input-output pairs. This iterative process is guided by a loss function that quantifies the difference between the predicted filtered output and the expected output, which is precomputed for comparison. The error is then backpropagated through the network, allowing the model to refine its parameters with each iteration. Optimization continues until a predefined number of epochs or a target loss threshold is reached. Once this criterion is satisfied, we calculate the Mean Squared Error (MSE) between the predicted and target filter coefficients, providing an objective measure of the estimation's accuracy. This iterative refinement process, combined with the neural network's ability to directly predict the filter coefficients, enables efficient and precise optimization. The proposed system architecture is detailed in Figure 1 and Algorithm 1, which outline the overall process and flow of optimization.

4.1. Introduction to Kolmogorov–Arnold Networks

Kolmogorov–Arnold Networks (KANs) are a new specialized class of neural networks designed to efficiently approximate complex, high-dimensional functions. Named after the Kolmogorov–Arnold representation theorem [23], these networks leverage fixed weights while training their activation functions (see Figure 0.1 from [20]). This design choice enhances both the interpretability and robustness of KANs in specific applications. Unlike traditional neural networks, such as Multi-Layer Perceptrons (MLPs), where only weights are learned during training, KANs offer a unique architecture that prioritizes symbolic representation and efficient computation [20]. KANs are particularly well-suited

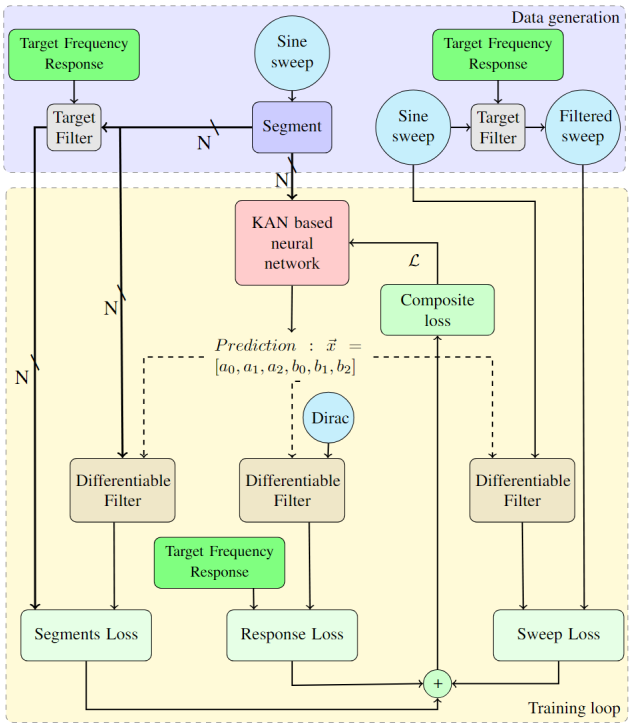

Figure 1: *Optimization system architecture.*

for tasks that require precise approximations of mathematical relationships, which makes them ideal candidates for filter coefficient estimation [21]. By leveraging their inherent adaptability and interpretability, KANs can provide a favorable balance between accuracy and computational efficiency, especially when predicting biquad filter coefficients. Furthermore, KANs integrate seamlessly into modern deep learning frameworks due to their differentiable nature, enabling the use of standard backpropagation algorithms. This differentiability allows for end-to-end optimization of complex systems that incorporate IIR filters, thereby improving overall system performance. Recent studies have highlighted the potential of KANs in symbolic formula representation [21, 20] and other domains where interpretability and computational efficiency are critical. Motivated by KANs properties, we explore their use in signal processing tasks by employing KANs for biquad filter design based on the frequency response. This approach not only capitalizes on the strengths of KANs in terms of approximation and interpretability but also enables the creation of adaptive and scalable filter design solutions, making them suitable for real-time applications.

4.2. Data Generation

To implement this gradient-based optimization approach, two distinct types of data are required:

- Target IIR filter coefficients data.
- Training input and output signals data.

4.2.1. Target filter coefficients data

To train and evaluate the proposed approach, a well-defined set of target filter coefficients is required. Following the methodol-

ogy presented in [13], we generate these coefficients by randomly sampling stable second-order polynomial coefficients for the biquad transfer function (Eq. (1)). The sampling process is carefully constrained to ensure the resulting filters are both valid and stable, preventing the inclusion of non-causal or unstable filters. Specifically, we enforce stability by ensuring that the poles lie strictly within the unit circle, as discussed in Section 4.3.

4.2.2. Training input and output signals data

For the optimization process to converge effectively, appropriate input data is necessary. In line with the approach outlined in [13], we employ an input signal and its corresponding filtered output signal within the optimization loop. Specifically, a long sine sweep signal (covers the whole frequency spectrum) is generated, which is then passed through the target biquad filter to obtain the filtered output. The resulting input-output time series pair is then resampled into multiple sequences of length N, which are used as input-output pairs for training. These input-output pairs are fed into the neural network iteratively until the filter coefficients converge to their optimal values.

4.3. Constraints on Coefficients Search Space

To enhance the computational efficiency and convergence of our KAN-based optimization, we introduce constraints that limit the search space for biquad coefficients. These constraints are derived from prior knowledge of the expected parameter ranges and application-specific requirements. By narrowing the search space, we ensure that only valid regions are sampled, which results in stable filters, as illustrated in Figure 1 in [24]. This reduction in the parameter space accelerates the optimization process while preserving accuracy.

In the case of the BA coefficients representation, we assume $a_0 = 1$, therefore the searched coefficients vector is shortened (n=5) to $x = [b_0, b_1, b_2, a_1, a_2]$. This formulation can be extended to a cascade of biquads to compute higher order IIR filter, where the search matrix is represented as

$$\mathbf{X} = \begin{bmatrix} b_{0,1} & b_{1,1} & b_{2,1} & a_{1,1} & a_{2,1} \\ & & \vdots & & \\ b_{0,k} & b_{1,k} & b_{2,k} & a_{1,k} & a_{2,k} \end{bmatrix} \tag{7}$$

where k denotes the index of the k-th biquad in the cascade of biquads. To maintain filter stability, the following constraints derived from [24] must be enforced during the search :

$$\begin{aligned} a_{1,k} &\leftarrow -2 \cdot \tanh(a_{1,k}) \\ a_{2,k} &\leftarrow \frac{(2 - |a_{1,k}|) \cdot \tanh(a_{2,k}) + |a_{1,k}|}{2} \end{aligned} \tag{8}$$

In the case of the Zero-Pole-Gain (ZPK) representation, we apply the following equivalent update conditions [8]. Moreover, to ensure numerical stability, a transformation is applied to the zeros and poles using the hyperbolic tangent function $\tanh(x)$, as shown below:

$$z' = \frac{(1-\epsilon) \cdot z \cdot \tanh(|z|)}{|z| + \epsilon}, p' = \frac{(1-\epsilon) \cdot p \cdot \tanh(|p|)}{|p| + \epsilon} \tag{9}$$

where ϵ is a small constant (e.g., $\epsilon = 10^{-8}$) to prevent division by zero. This transformation ensures that the values of the zeros and poles are bounded, maintaining numerical stability, especially for very small or very large values of z and p [8].

4.4. Loss Functions

To integrate IIR structures with KANs within a machine learning framework, it is essential to utilize platforms such as PyTorch or TensorFlow [25, 26], which support automatic differentiation of computational graphs. The ability to compute gradients automatically facilitates the use of custom loss functions, enabling seamless network training. In this context, the primary consideration is to ensure that the loss function is differentiable. This requirement allows the backpropagation process to effectively update the model parameters, enabling efficient optimization of the network. In our study, we explore three loss functions:

- **Impulse Response Loss**
- **Filtered Sweep Loss**
- **Filtered Sweep Segments Loss**

These loss functions provide effective means to optimize the filter's behavior for both spectral characteristics (frequency response) and temporal characteristics (filtered sweep) and can be used separately or combined.

4.4.1. Impulse Response Loss

The *Impulse Response Loss* function minimizes the difference between predicted and target frequency responses, using the precomputed impulse response of the target as a reference. The predicted filter's frequency response is obtained by applying the filter to a Dirac delta function and performing a Fast Fourier Transform (FFT). Consider $h(t)$ the impulse response of the filter, which is obtained by applying the predicted coefficients to a Dirac delta function $\delta(t)$ in $t - domain$. This leads to the frequency response $H(\omega) = \text{FFT}(h(t))$. Hence, the magnitude and phase of the frequency response are:

$$\begin{aligned} |H(\omega)| &= 20\dot{\log}_{10}(|H(\omega)| + \epsilon) \\ \angle(H(\omega)) &= \text{angle}(H(\omega)) \end{aligned} \tag{10}$$

where ϵ is a small constant to prevent numerical issues. The target frequency response $\mathbf{y}_{\text{target}}$ is the concatenation of the magnitude and phase responses of the desired filter. The predicted frequency response $\mathbf{y}_{\text{pred}}$ is obtained by the same process. The response loss function is then defined as the Mean Squared Error (MSE) between the predicted and target frequency responses:

$$\mathcal{L}_{\text{impulse}} = \text{MSE}(\mathbf{H}(\omega)_{\text{pred}}, \mathbf{H}(\omega)_{\text{target}}) \tag{11}$$

This loss ensures that the frequency response of the predicted filter matches the desired response in both magnitude and phase.

4.4.2. Filtered Sweep Loss

The *Expected Filtered Sweep-based Loss* assesses the difference between the predicted output and the target output after filtering a sinusoidal sweep signal. This qualifies the filter's accuracy when applied for a defined range of frequencies [13]. Given an input

[1] We use Pytorch (https://pytorch.org/) in our experiment

signal $x(t)$ (e.g., a sweep), the output signal $y(t)$ after applying the filter with coefficients **sos** (second-order sections) is given by:

$$y(t) = \text{filter}(x(t), \mathbf{sos}) \tag{12}$$

where the filter operation is performed using the **s**econd-**o**rder **s**ections **sos** (if the reference is provided during testing) or by multiplying $x(t)$ in the frequency domain with the target filter impulse response. The loss is then calculated using the MSE between the predicted output $y_{\text{predicted}}(t)$ and the target output $y_{\text{target}}(t)$:

$$\mathcal{L}_{\text{filtered\_sweep}} = \text{MSE}(y_{\text{predicted}}(t), y_{\text{target}}(t)) \tag{13}$$

4.4.3. *Filtered Sweep Segments Loss*

Similar to the *Filtered Sweep Loss*, this function evaluates the difference between the predicted and target outputs after filtering N overlapping segments from a sinusoidal sweep signal. This is given by:

$$\mathcal{L}_{\text{filtered\_segments}} = \sum_{n=1}^{N} \text{MSE}(y_{\text{predicted\_seg[n]}}(t), y_{\text{target\_seg[n]}}(t)) \tag{14}$$

4.5. Evaluation

Upon completion of the optimization process—either after a sufficient number of epochs or upon reaching a predefined loss threshold—the optimization loop terminates. The predicted filter coefficients are then used to compute the filter's impulse response. This response is subsequently employed to generate the magnitude and phase plots, which are compared to those of the target filter. Figure 2 shows the magnitude and phase responses of the predicted filter versus the target, along with the corresponding differences. The zeros and poles of the predicted filter are visualized in the polar coordinate system and analyzed relative to the target configuration, as shown in Figure 3. Finally, the loss curve is examined to assess its smoothness—providing insight into the convexity of the optimization landscape—and to evaluate the overall effectiveness of the optimization process.

5. RESULTS

This section presents the outcome of the experiments conducted to assess the performance of the proposed KAN-based optimization for IIR filters design. The evaluation was carried out using a set of 10 filter designs, with a variety of hyperparameters tested for each method. Furthermore, we compared the performance of the KAN-based approach with two alternative methods: an optimization method based on RNNs, as described in [13], and a neural network-based approach utilizing MLPs, as outlined in [8].

5.1. RNN-based Approach

The RNN-based approach demonstrated strong stability and accuracy across most test cases [13], but its convergence times were significantly longer, primarily due to the time step-based nature of the method. As a result, RNN proved to be less efficient than the KAN-based method, especially in applications that require rapid results. Although the RNN maintained stable performance across various filter designs, it occasionally faced challenges in convergence, often becoming trapped in local minima. One potential strategy to address this issue is to initialize the optimization with different starting points, thus mitigating the risk of local minima and improving convergence reliability [27].

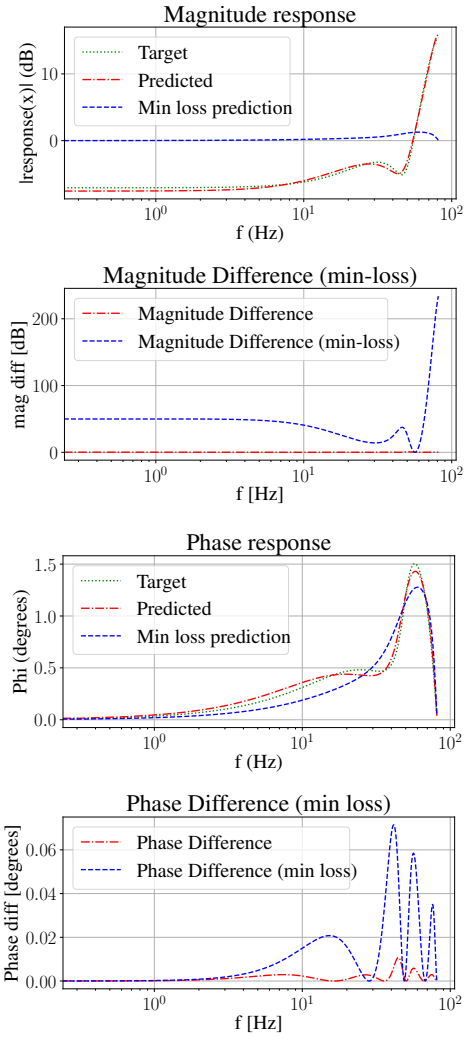

Figure 2: *Comparison of magnitude (Top) and phase (bottom) responses of target filter (solid line) vs. predicted filter (dashed line) and minimum-loss prediction (dotted line). Each plot has a corresponding magnitude and phase difference between target and predicted outputs.*

5.2. MLP-based Approach

The MLP-based approach initially demonstrated potential, but struggled to consistently generate accurate IIR filter designs. Although it frequently approximated the target frequency response, it rarely achieved an exact match, likely due to overfitting or discrepancies between the training and test data. A significant limitation encountered in our tests was the model's inability to design IIR filters of order lower than 4 or adjust to those designs, preventing a meaningful comparison between different filter orders. Performance also degraded substantially when the approach was applied to higher-order filters, highlighting its scalability issues. Despite the practical framework suggested by Colonel et al. [8], the MLP approach requires large datasets and extensive training time to perform effectively. Furthermore, once trained, the

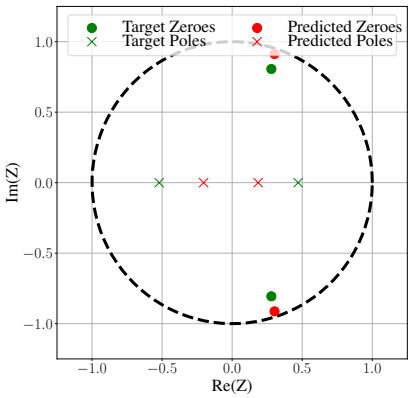

Figure 3: *Poles and zeros in the Z-plane for the target biquad (green) and the KAN-predicted filter (red) within the stability region (unit circle).*

model's performance on unseen data proved difficult to diagnose, especially when the data diverged from the training set. The inherent "black-box" nature of MLPs further complicates troubleshooting, necessitating more representative training data or substantial architectural modifications, both of which are time-consuming and resource-intensive tasks.

5.3. KAN-based Optimization

The KAN-based method consistently exhibited stable performance and faster convergence relative to the RNN-based approach. The loss function showed a faster reduction during training. However, similar to the RNN approach, the loss occasionally oscillated, likely due to challenges in hyperparameter tuning or suboptimal initial conditions. Although a search solution akin to the one cited by Kirkpatrick et al. [27] could be applied to mitigate these fluctuations, the primary advantage of the KAN-based method lies in its faster processing speed. This speed enables more parallel runs and, consequently, a higher number of restarts within a given time frame. Despite occasional oscillations, the KAN-based approach consistently demonstrated strong performance in terms of both speed and loss convergence.

To optimize the hyperparameters, we used the Optuna framework [28], exploring various configurations with the goal of minimizing the total loss. The optimization process involved 1500 trials, each with a specific set of hyperparameters. The results were manually analyzed and the optimal configurations were selected to construct Figure 4, which highlights the desired ranges for the hyperparameters to ensure convergence and perfect predictions [2]. Furthermore, we examined a composite loss function combining filtering and frequency response losses: $\mathcal{L} = \mathcal{L}_{\text{filtered\_sweep}} + \mathcal{L}_{\text{impulse}} + \mathcal{L}_{\text{filtered\_segments}}$. Our experiments revealed that the filtering-based loss functions outperformed the impulse-based one. The plot in Figure 5 illustrates that minimizing the loss function should lead to correct predictions. However, as shown in the density distribution, some high-loss cases still resulted in correct predictions.

[2] Perfect predictions consist of final results where the predicted and target responses perfectly overlap. These were manually selected.

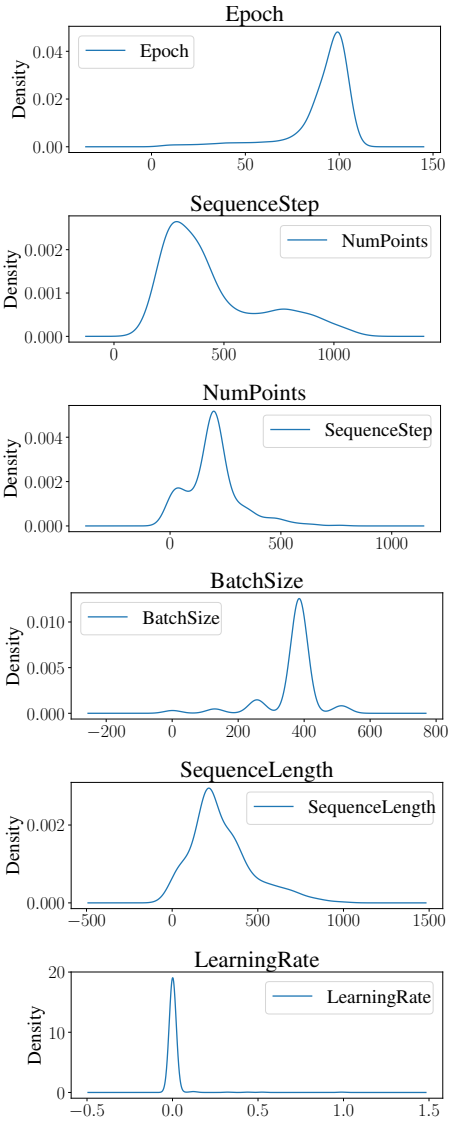

Figure 4: *Distributions of hyperparameters (from Optuna trials) that yielded exact convergence ("perfect" predictions where output matched target).*

5.4. Summary of Results

To summarize the key findings, the KAN-based method outperformed both the RNN and MLP approaches. It excelled over the RNN in terms of both speed and accuracy, while also offering a more interpretable and debuggable framework compared to the MLP. The RNN demonstrated stability, but was slower, whereas the MLP struggled to produce accurate predictions for targets that are not modeled in the training data. A more detailed comparison is provided in Table 1.

5.5. Discussion

The KAN-based approach offers significant advantages in computational efficiency and stability compared to gradient-based and neural network methods, making it particularly suitable for filter design and parametric estimation tasks. It achieves faster conver-

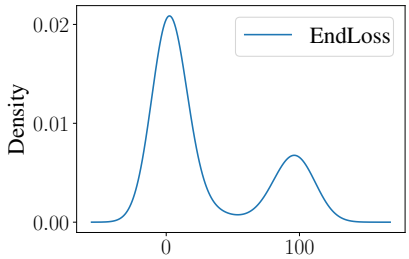

Figure 5: *Final loss distribution for perfect predictions.*

Method / Metric	KAN-based	RNN-based [13]	MLP-based [8]
Type	Gradient based optimization	Gradient based optimization	Trained network
Training duration (Epochs)	Fastest training loop	Fast training loop	Slow (requires a lot of training data & lengthy training)
Prediction duration (Epochs)	Moderate	Slowest	Fastest (requires lengthy training)
Accuracy	High	High	Moderate to low
Loss	Smooth loss (with some exceptions)	Smooth loss (with many exceptions due to RNN nature)	Data dependent

Table 1: *Comparison of filter coefficients approaches.*

gence and smoother loss reduction than the RNN-based method, which, although stable, requires longer training times and is more prone to local minima. In contrast to the MLP-based method, the KAN approach is more flexible and adaptable, providing a more practical solution, especially in scenarios requiring rapid optimization. Although the KAN method demonstrates strong performance, it is still in the early stages of development. One of the main challenges is handling non-differentiable representations for higher-order filters, which currently limits its applicability. This limitation arises from the absence of a stable differentiable implementation (based on a cascade of biquads) for high-order IIR filters. Expanding the method to accommodate higher-order filters and evaluating it across diverse datasets are critical to refining its capabilities and improving generalization. Moreover, while the filtering-based loss function outperforms the impulse-based one, further research is needed to identify the optimal loss function for filter design.

The use of Optuna for hyperparameter optimization has enhanced the method's efficiency. However, tuning remains challenging, particularly for achieving consistent results across different filter types, necessitating further experimentation. Despite these challenges, the KAN-based approach offers a robust, flexible, and efficient alternative to RNNs and MLPs, providing faster processing and improved performance stability.

6. CONCLUSIONS

In this paper, we introduced a KAN-based optimization method for estimating the coefficients of second-order IIR filters (biquad). The approach offers a viable alternative to traditional gradient-based and neural network methods for filter design and parametric estimation tasks. It demonstrated faster convergence and greater stability compared to RNN-based methods and greater flexibility than MLP-based approaches. Although the method shows significant potential, it is still in the early stages and has challenges to address. Further testing and extension to higher-order filters are necessary to fully realize its capabilities.

7. REFERENCES

[1] Vesa Välimäki and Joshua D. Reiss, "All about audio equalization: Solutions and frontiers," *Applied Sciences*, vol. 6, no. 5, pp. 129, 2016.

[2] A. Martinez-Leira, R. Vicen-Bueno, R. Gil-Pita, and M. Rosa-Zurera, "Acoustic feedback reduction based on fir and iir adaptive filters in ite digital hearing aids," in *Proc. 2008 IEEE Int. Conf. on Audio, Language and Image Processing (ICALIP)*, Shanghai, China, July 2008, pp. 1442–1448.

[3] Eero-Pekka Damskägg, Lauri Juvela, Etienne Thuillier, and Vesa Välimäki, "Deep learning for tube amplifier emulation," in *Proc. 2019 IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, Brighton, UK, May 2019, pp. 471–475.

[4] Y. Chan and R. Langford, "Spectral estimation via the high-order yule-walker equations," *IEEE Trans. Acoustics, Speech and Signal Process.*, vol. 30, no. 5, pp. 689–698, 1982.

[5] M. C. Lang, "Weighted least squares iir filter design with arbitrary magnitude and phase responses and specified stability margin," in *Proc. 1998 IEEE Symp. on Advances in Digital Filtering and Signal Processing (ADFSP)*, 1998, pp. 82–86.

[6] V. L. Stonick and D. Massie, "Optimal ls iir filter design for music analysis/synthesis," in *Proc. 1992 IEEE Int. Symp. on Circuits and Systems (ISCAS)*, San Diego, CA, USA, May 1992, pp. 2405–2408.

[7] S. Nishimura and Hai-Yun Jiang, "Gradient-based complex adaptive iir notch filters for frequency estimation," in *Proc. Asia Pacific Conf. on Circuits and Systems (APCCAS)*, 1996, pp. 235–238.

[8] Joseph T. Colonel, Christian J. Steinmetz, Marcus Michelen, and Joshua D. Reiss, "Direct design of biquad filter cascades with deep learning by sampling random polynomials," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Singapore, May 2022, pp. 3104–3108.

[9] Shahan Nercessian, "Neural parametric equalizer matching using differentiable biquads," in *Proc. Digital Audio Effects (DAFx-20)*, Vienna, Austria, Sept. 2020, pp. 265–272.

[10] Jesse Engel, Lamtharn Hantrakul, Chenjie Gu, and Adam Roberts, "Ddsp: Differentiable digital signal processing," in *Proc. 8th Int. Conf. on Learning Representations (ICLR 2020)*, Addis Ababa, Ethiopia, Apr. 2020.

[11] Yoshiki Masuyama, Gordon Wichern, François G. Germain, Zexu Pan, Sameer Khurana, Chiori Hori, and Jonathan Le Roux, "NIIRF: Neural IIR filter field for HRTF upsampling and personalization," in *Proc. 2024 IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, Seoul, Republic of Korea, Apr. 2024.

[12] Alec Wright, Eero-Pekka Damskägg, and Vesa Välimäki, "Real-time black-box modelling with recurrent neural networks," in *Proc. 22nd Int. Conf. Digital Audio Effects (DAFx-19)*, Birmingham, UK, Sept. 2019, pp. 248–255.

[13] Boris Kuznetsov, Julian D. Parker, and Fabián Esqueda, "Differentiable iir filters for machine learning applications," in *Proc. Digital Audio Effects (DAFx-20)*, Vienna, Austria, Sept. 2020, pp. 297–303.

[14] Giovanni Pepe, Leonardo Gabrielli, Stefano Squartini, and Luca Cattani, "Designing audio equalization filters by deep neural networks," *Applied Sciences*, vol. 10, no. 7, pp. 2483, 2020.

[15] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio, "On the difficulty of training recurrent neural networks," in *Proc. 30th Int. Conf. on Machine Learning (ICML)*, Atlanta, GA, USA, June 2013, pp. 1310–1318.

[16] António H. Ribeiro, Koen Tiels, Luis A. Aguirre, and Thomas B. Schön, "Beyond exploding and vanishing gradients: analysing rnn training using attractors and smoothness," in *Proc. 23rd Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, Aug. 2020, pp. 2370–2380.

[17] Purbaditya Bhattacharya, Patrick Nowak, and Udo Zölzer, "Optimization of cascaded parametric peak and shelving filters with backpropagation algorithm," in *Proc. 23rd Int. Conf. Digital Audio Effects (DAFx-20)*, Vienna, Austria, Sept. 2020, pp. 101–108.

[18] Anders R. Bargum, Stefania Serafin, Cumhur Erkut, and Julian D. Parker, "Differentiable allpass filters for phase response estimation and automatic signal alignment," in *Proc. Digital Audio Effects (DAFx-23)*, Copenhagen, Denmark, Sept. 2023, pp. 235–242.

[19] Chin-Yun Yu, Christopher Mitcheltree, Alistair Carson, Stefan Bilbao, Joshua D. Reiss, and György Fazekas, "Differentiable all-pole filters for time-varying audio systems," in *Proc. Digital Audio Effects (DAFx-24)*, Guildford, UK, Sept. 2024, pp. 345–352.

[20] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljacic, Thomas Y. Hou, and Max Tegmark, "Kan: Kolmogorov–arnold networks," in *Proc. International Conference on Learning Representations (ICLR)*, 2025, pp. 70367–70413.

[21] Runpeng Yu, Weihao Yu, and Xinchao Wang, "Kan or mlp: A fairer comparison," *arXiv preprint*, vol. 2407.16674, 2024.

[22] Florian Mockenhaupt, Joscha Simon Rieber, and Shahan Nercessian, "Automatic equalization for individual instrument tracks using convolutional neural networks," in *Proc. Digital Audio Effects (DAFx-24)*, Guildford, UK, Sept. 2024, pp. 57–64.

[23] Tianrui Ji, Yuntian Hou, and Di Zhang, "A comprehensive survey on kolmogorov arnold networks (kan)," *arXiv preprint*, vol. 2407.11075, 2024.

[24] Shahan C. Nercessian, Andy M. Sarroff, and Kurt James Werner, "Lightweight and interpretable neural modeling of an audio distortion effect using hyperconditioned differentiable biquads," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Toronto, Canada, June 2021, pp. 890–894.

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems (NeurIPS 2019)*, Vancouver, BC, Canada, Dec. 2019, vol. 32, pp. 8024–8035.

[26] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng, "Tensorflow: A system for large-scale machine learning," in *Proc. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Savannah, GA, USA, Nov. 2016, pp. 265–283.

[27] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[28] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. 25th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, Anchorage, AK, USA, Aug. 2019, pp. 2623–2631.

8. APPENDIX

Algorithm 1: KAN-based Optimization for IIR Filter Design

Data: Target filter frequency response,
input sweep signal, and expected filtered signal
Result: Optimized IIR filter coefficients using KANs

Initialize:
model ← KAN_based_network() ;
optimizer ← Adam($lr = 1e-3$) ;
L_{seg} ← FilterSegmentsLoss();
L_{resp} ← ResponseLoss();
L_{sweep} ← FilterSweepLoss();
training_data ← segment(sweep, filtered_sweep);
for *epoch* ***in*** *1 to n_epochs* **do**
    epoch_loss ← 0;
    for *inputs, targets* ***in*** *training_data* **do**
        pred_coeffs ← model.forward(inputs);
        pred_filtered_sweep ← differentiable_iir(pred_coeffs, sweep);
        outputs ← differentiable_iir(pred_coeffs, inputs);
        $\ell_1 \leftarrow L_{\text{seg}}$(outputs, targets);
        $\ell_2 \leftarrow L_{\text{resp}}$(pred_coeffs, target_freq_response);
        $\ell_3 \leftarrow L_{\text{sweep}}$(filtered_sweep, pred_filtered_sweep);
        loss ← $\ell_1 + \ell_2 + \ell_3$;
        Store loss for tracking;
    optimizer.zero_grad();
    loss.backward();
    optimizer.step();
After training:
Compute poles, zeros, and gain from target ;
Evaluate the results using MSE and polar coordinates;