# Hierarchical Predictive Coding and Interpretable Audio Analysis-Synthesis

André Ofner, Johannes Schleiss and Sebastian Stober

Otto von Guericke University, Magdeburg, Germany
{ofner,schleiss,stober}@ovgu.de

**Abstract.** Humans efficiently extract relevant information from complex auditory stimuli. Oftentimes, the interpretation of the signal is ambiguous and musical meaning is derived from the subjective context. Predictive processing interpretations of brain function describe subjective music experience driven by hierarchical precision-weighted expectations. There is still a lack of efficient and structurally interpretable machine learning models operating on audio featuring such biological plausibility. We therefore propose a bio-plausible predictive coding model that analyses auditory signals in comparison to a continuously updated differentiable generative model. For this, we discuss and build upon the connections between Infinite Impulse Response filters, Kalman filters, and the inference in predictive coding with local update rules. Our results show that such gradient-based predictive coding is useful for classical digital signal processing applications like audio filtering. We test the model capability on beat tracking and audio filtering tasks and conclude by showing how top-down expectations modulate the activity on lower layers during prediction.

**Keywords:** Predictive Processing, Machine learning, Digital Signal Processing

## 1 Introduction

### 1.1 Audio Processing and Predictive Coding in the Human Brain

Research on human auditory processing has demonstrated that humans are efficient at tracking stochastic auditory regularities and can even disentangle stationary parts, e.g. fundamental frequencies, from dynamic transformations, e.g. resonances, in musical events. The predictive coding (PC) theory is a popular framework in neuroscience that explains how such complex human processing could arise from a relatively simple repeated algorithmic pattern implemented in neurons, namely the reduction of prediction errors [1, 2]. Recent advances in machine learning have progressed towards predictive coding models that update simulated neurons with errors computed local to these neurons, in contrast to the backpropagation through entire neural networks that drive most current deep learning systems [3]. Through the use of local errors and simple neural operations (e.g. summation or addition) PC networks are plausible models of the computations in biological neurons. From an engineering perspective, predictive coding networks (PCN) with a single layer already deliver useful computations, like the source-filter separation in Linear Predictive Coding (LPC), a widely used Digital Signal Processing (DSP) method. To live up to their full potential, PCNs need hierarchical structure. In hierarchical PCNs hidden layers predict the expected latent states of lower layers. However, there is still a

lack of hierarchical and biologically plausible machine learning models that combine the possibility to operate on raw audio with reasonable performance on classical DSP tasks. These tasks can include audio filtering or extracting musical information, e.g. beat timings, from audio.

### 1.2 Hierarchical Predictive Coding and Digital Signal Processing

The number of existing studies employing predictive coding to process raw audio is limited and available methods are generally difficult to interpret. Moreover, PC models in neuroscience are generally restricted to simple auditory stimuli or even symbolic inputs [4, 5]. Still, there are similarities between the structures of Infinite Impulse Response (IIR) filters and recurrent neural networks (RNN), classes that are already widely used in DSP applications and those models that model human (auditory) cognition more specifically, in particular the Kalman filter or predictive coding networks. These connections will be discussed in more detail in Section 2.

A major challenge when employing predictive coding networks for engineering tasks is that they only deliver approximate results during learning and inference. This poses a major drawback in the context of DSP tasks, where high accuracy is generally required. Furthermore, it is difficult to design efficiently operating hierarchical PC models, which would have the advantage of naturally scaling to larger DSP systems with meaningful cognitive interpretations. To solve these challenges, we resort to the structural similarities between PC models and established DSP methods in the next section and then introduce a hierarchical PC model [1].

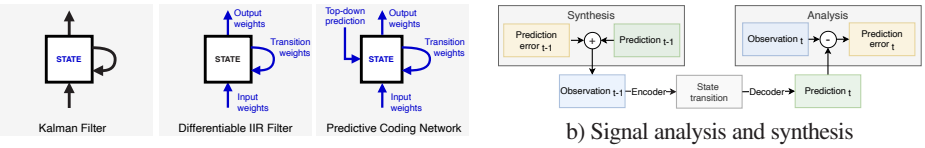
## 2 Related Work

The similarity between IIR filters, Kalman filters, RNNs, and predictive coding networks is particularly apparent when one views these models in their state-space (SSM) form. Figure 1 a) provides an overview of these related classes in state-space form, such as they are used in tasks typical for each class. Aspects of learned model structure, such as filter coefficients, are referred to as weights in the context of artificial networks. Generally speaking, ”inference” refers to employing these given coefficients (i.e. weights) to update hidden representations, while ”learning” refers to the slower process of optimizing weights.

While the signal flow of the model classes is directly comparable, differences arise in the way inference and learning are addressed in typical tasks. Kalman filters are usually used for dynamic inference given prior assumptions on the data, resulting in mathematically exact updates of their latent state. The deterministic class of IIR filters is typically used to apply a previously designed transfer function to incoming signals, where output signals are a weighted combination of previously processed signals. Some exceptions, such as differentiable IIR filters allow to learn weights during application [6]. Kalman filters and predictive coding networks are typically modeled as probabilistic generative models, keeping track of an inferred latent state with associated variance (or inverse precision). Both have found applications in modeling cognitive and neural processes. In contrast to Kalman filters, optimization in predictive coding networks generally addresses state inference and weights learning simultaneously.

Finally, PCNs can include internal predictions of their latent states, i.e. ”top-down” expectations about activities in lower PCN layers [2, 7]. This hierarchical structure is similar,

[1] Code is available at github.com/andreofner/APC




a) Comparison in state-space form

b) Signal analysis and synthesis with autoregressive predictive coding

**Fig. 1.** a) Comparison of Kalman filters, differentiable IIR filters, and gradient-based predictive coding networks in state-space form. Blue color indicates variables that are optimized in a typical filtering application for each model. b) Signal analysis and synthesis with autoregressive predictive coding and linear activation functions: In the analysis stage, observations at time-step $t$ are mapped to hidden states using encoder weights. The learned transition dynamics are then applied to the latent state. Outgoing predictions for the next timestep $t+1$ are computed via decoder weights that map from the updated latent state to the expected sensory input. During synthesis, the prediction error is fed to the model jointly with the previous prediction.

but not identical, to the multi-layer architecture of deep neural networks, which typically lack the feedback connections that are inherent to PCNs. More specifically, DNNs can be interpreted as corresponding to pyramidal dendritic connections in the biological counterpart. This means that DNNs, possibly with multiple layers, connect adjacent variables in PCN layers [8]. Finally, existing work on PCN architectures has explored "dynamical" predictive coding, where not only the activity of lower layers but also (multiple) temporal derivatives are modelled [9]. Here, we explore the audio DSP capabilities of single-layer and hierarchical PCN models interpreted as biologically plausible Neural Kalman filters. This PCN class has been discussed for single-layer models in [10].

### 2.1 Autoregressive Signal Filtering with State-Space Models

Signal analysis with autoregressive filters at discrete time-steps $t$ can be described with respect to a steady state transfer function $H(z)$

$$H(z) = \frac{G}{1 - \sum_{j=1}^{k} a_j z^{-j}} = \frac{G}{A(z)} \tag{1}$$

with input gain $G$ [11, 12]. The parameters $a_j$ with $1 \leq j \leq k$ and $G$ of this state transfer function can be optimized with respect to the prediction error $e(x)$ between predicted signal $p(t)$ and observed signal $o(t)$, also referred to as excitation or residual signal:

$$e(t) = \frac{1}{G}(o(t) - \sum_{j=1}^{k} a_j o(t-j)) \tag{2}$$

The SSM of this generalized prediction error filter is updated with the following difference equation:

$$\begin{aligned} z[t+1] &= A[t]z[t] \\ o[t] &= C[t]z[t] \end{aligned} \tag{3}$$

where $z[t]$ is the state vector at timestep $t$ and the prediction coefficients $a_j$ are represented by weights $A$ and $C$. All four discussed model classes, despite originating from the different

fields can be interpreted in prediction error minimizing SSM form. Linear predictive coding (LPC), a widely used DSP tool, draws from this possibility for the design of IIR coefficients. LPC is typically used for signal compression, particularly for speech coding, by separating stationary residual signals from imposed resonances [13]. This theoretically allows to analyse and synthesize signals using the same model. However, the efficient algorithms employed in LPC are not directly biologically interpretable and generally do not actually use a SSM to find the coefficients. From this perspective, our work generalises LPC towards the more general class of hierarchical PCN, where analysis and synthesis use the same model.

**RNN and Differentiable IIR Filter** Recurrent neural networks, in their simplest form, can be expressed by the following difference equations [6, 14]:

$$\begin{aligned} z[t+1] &= \sigma_z(W_z z[t] + U_z x[t+1] + b_z) \\ y[t+1] &= \sigma_y(W_y z[t+1] + b_y) \end{aligned} \tag{4}$$

with hidden states $z$, inputs $x$ and outputs $y$. $W$ and $U$ are trainable weights and $b$ are biases. Known from previous work is that, in the case where activation functions $\sigma$ are (non-)linear and the biases are set to zero, this structure directly resembles a (non-)linear all-pole IIR filter

$$\begin{aligned} z[t+1] &= W_z z[t] + U_z x[t+1] \\ y[t+1] &= W_y z[t+1] \end{aligned} \tag{5}$$

which scales to arbitrary order of transfer functions $H(z)$ (also referred to as the filter order) and allows to train differentiable IIR filters using the optimization methodology for RNNs [6]. A useful generalized state space form for such IIR filters is

$$\begin{aligned} z[t+1] &= Az[t] + Bx[t] \\ y[t+1] &= Cz[t+1] + Dx[t+1] \end{aligned} \tag{6}$$

where matrices $A$,$C$ represent the learnable weights for latent state transition and output transformation and $B$,$D$ are weights for input transformations [6].

**Kalman Filters** The Kalman filter gained large popularity in fields such as engineering, statistics, and neuroscience and filters data points with respect to a probabilistic latent state and their expected precision. Typically, dynamics and observation models are linear and the observed noise and the latent states are modeled as Gaussian distributions. Similar to the previously discussed model classes, the Kalman filter can be described in SSM form:

$$\begin{aligned} z[t+1] &= Az[t] + Bu[t] + v \\ y[t+1] &= Cz[t+1] + w[t] \end{aligned} \tag{7}$$

with hidden states $h_t$ at discrete timesteps $t$. Correspondingly to the deterministic IIR filter, the weights of the transition matrix $A$ describe the linear dynamics. The weights of matrix B and C parameterize the observation model. Weights $B$ transform the control inputs $u$, i.e. known inputs to the system and $C$ map from inferred state to the sensory prediction. Finally, $v$ and $w$ are white noise Gaussian processes with mean zero. The Gaussian prior $p(z_{t+1})$ and posterior distribution $p(z_{t+1} \,|\, y_{1\ldots t}, x_t)$ of the Kalman filter are parameterized by their sufficient statistics, the mean $\mu$ and covariance matrix $\Sigma_z$ [10, 15].

**Gradient-Based Predictive Coding** Gradient-based predictive coding, as described in has been applied to an approximation of the exact inference in the Kalman filter [10]. In the simplest case, without observations or control inputs, we have a state space model of the form

$$\begin{aligned} z[t+1] &= Az[t] \\ y[t+1] &= Hz[t+1] \end{aligned} \tag{8}$$

where $A$ and $H$ are learnable matrices for the state transition dynamics and the observation model respectively.

Following [10], we define the loss function of the predictive coding filter as:

$$argmin_{\mu_{t+1}} L = argmax_{\mu_{t+1}} p(y_{t+1} \,|\, z_{t+1}) p(z_{t+1} \,|\, z_t) \tag{9}$$

In this formulation, weights $A$ and $H$ and the inferred hidden state $z$ (or, more specifically, its mean $\mu$ and variance $\epsilon_z$ parameters) can be updated using gradient descend based on the precision weighted prediction errors local to the layer [10]:

$$\frac{dL}{d\mu_{t+1}} = -H^T \Sigma_z \epsilon_z + \Sigma_x \epsilon_x, \quad \frac{dL}{dA} = -\Sigma_x \epsilon_x \mu_t^T, \quad \frac{dL}{dC} = -\epsilon_y \mu_{t+1}^T \tag{10}$$

with sensory prediction errors $\epsilon_y = y - H\mu_{t+1}$ and state prediction errors $\epsilon_z = \mu_{t+1} - A\mu_t$ [10]. Intuitively speaking, this means that each layer optimizes the quality of its signal predictions $p_{y_{t+1}} = H\mu_{t+1}$ and of its state predictions $p_{\mu_{t+1}} = A\mu_t$. As this optimization process happens locally informed and in parallel for each optimized variable, many different possible outcomes decrease the prediction error. E.g., quickly adapting observation weights $H$ induce different latent states than a slowly optimized observation model. Similarly, missing accuracy in the observation model might be compensated by hidden state optimization.

A more general form of the predictive coding SSM includes additional weights for control inputs $u$ and observed inputs $x$:

$$\begin{aligned} z[t+1] &= Az[t] + Bu[t] \\ y[t+1] &= Hz[t+1] + Dx[t] \end{aligned} \tag{11}$$

In summary, we see that single layer predictive coding models and Kalman filters can be represented using the same SSM as IIRs and RNNs (excluding nonlinearities), but additionally differentiate between control and observed inputs.

## 3 Hierarchical Predictive Coding of Audio

To create a hierarchy of layers with local computations, we can augment the predictive coding SSM mentioned in equation 11 with two sets of weights, $F$ and $G$. These weights modulate the influence of the layer's own latent state $z$ in comparison to a top-down prediction of this state $z_{td}$ provided by a higher layer:

$$\begin{aligned} z[t+1] &= FAz[t] + GAz_{td}[t] + Bu[t] \\ y[t+1] &= Hz[t+1] + Dx[t] \end{aligned} \tag{12}$$

and denote the weighted state prediction from current and next higher layer with $\hat{z} = Fz + Gz_{td}$. In all experiments, we ignore control inputs $u$, which could receive known

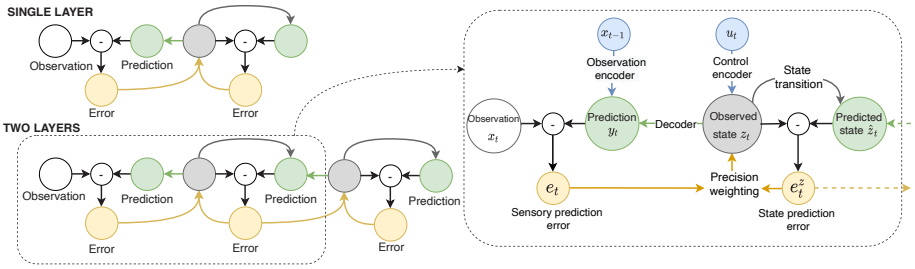



**Fig. 2.** Predictive Coding network for hierarchical Kalman filtering: At each timestep $t$, predictions $y_t$ are generated from a latent state $z_t$ using decoder weights that are optimized towards the sensory prediction error $e_t$ between observation $x$ and prediction $y$. Future latent states $z_{t+1}$ are computed with learnable transition weights. The transition weights are optimized towards the state prediction error $e_t^z$ between predicted state $\hat{z}_t$ and the next inferred state $z_t$. Hidden PC layers minimize the prediction error $e_t^z$ from a "top-down" prediction of the state. The hidden state $z$ is optimized towards sensory and state prediction error $e_t$ and $e_t^z$ and creates a balance between outgoing and incoming predictions. Optional encoders allow to predict with respect to past observations $x_{t-1}$ or control inputs $u$.

additional (action) signals and feed past observations $x_{t-1}$ to the observation encoder for the filtering task presented in section 4.3.

The state prediction error now includes the additional input and weights:

$$\epsilon_z = \mu[t+1] - FA\mu[t] - GA\mu_{td}[t] \tag{13}$$

Figure 2 shows an overview of a single layer predictive coding model and how multiple layers can be connected through locally informed predictions and prediction error signals. More precisely speaking, the lowest PC layer directly predicts audio inputs and receives prediction error $e_t$ at every timestep. In contrast, hidden PC layers predict the hidden latent states ("cause units") of the lower layer and receive state prediction error $e_t^z$. Both lowest and hidden PC layers additionally optimize the weights of their transition model that maps from currently inferred state $z_t$ to the next state $z_t+1$. We can interpret weights $F$ and $G$ as part of the prediction units that produce the optimal state predictions $z_{t+1}$ given the transition model $A$. Finally, the latent state $z_t+1$ is optimized in parallel via gradient descent to minimize the summed precision weighted prediction error $e_t+e_t^z$ local to the respective layer.

We use an overlap-and-add processing approach which is commonly used in DSP, meaning that the PCN processes audio signals in overlapping sequences. For all experiments, the lowest PCN layer processes these sequences sample-by-sample. Hidden layers have identical update frequencies. We found that sequence sizes between 16 and 2048 frames provide meaningful results. The hop-length was set to half the sequence length.

### 3.1 Audio Analysis and Synthesis with Predictive Coding

Assuming purely linear prediction and a well-trained model, using the PCN for audio resynthesis is possible by reverting the process that computes the residual signal at timestep $t$ (i.e. linear prediction error) from the prediction during analysis. Figure 1 b) shows an overview of the steps for synthesis and analysis given at the lowest layer of a hierarchical predictive

coding model. While this is not the only possible approach to analyze and synthesize signals with predictive coding networks, it has the advantage of relatively exactly replicating the approach taken in LPC. In LPC the coefficients minimizing the squared error during the linear prediction of the next sample resemble compressed versions of the resonances (typically formants in speech coding) and allow the signal to be transmitted with high compression rates through block-wise filter coefficients and down-sampled residual signals. For linear prediction, this LPC residual signal is equal to the prediction error that arises in (gradient-based) predictive coding.

Assuming linear PCN weights and audio with stationary parts, we expect that resonant parts of the audio are gradually removed from the residual. Added hierarchy and non-linear activations will affect the meaning of the first layer's residual signal, e.g. through emerging attentional processes.

## 4 Results

### 4.1 Beat Tracking

In order to quantitatively assess the possibility to extract music information from raw audio using prediction errors, we resort to a beat tracking task using two datasets: The SMC MIREX dataset is commonly used for beat tracking evaluation [16]. Our second evaluation is based on finger tapping recordings in the NMED-T dataset that focuses on electroencephalographic (EEG) recordings during music perception [17]. We choose an approach similar to the predominant local pulse (PLP) method described in Grosche et al. [18] and predict beat timings based on a local enhancement of a novelty function. The novelty function in [18] is based on spectral flux, the spectral difference between subsequent Fourier transformed audio inputs. We feed Fourier transformed audio inputs to the PCN (this being the only place where the PCN inputs are not audio samples) and use the prediction error from a single layer PCN to compute the novelty curve. Wherever possible, we use the same FFT parameters as used in Grosche et al. [18] but do not tune any other hyper parameters. For comparison to other approaches, we report the F-measure and two continuity-based metrics: CMLt, measuring correctly tracked beats at the metrical level, and AMLt, which allows variations such as double, half or offbeat variations [19]. All evaluations are based on the mir_eval package [20]. Next to the PLP model, we compare our approach to established baselines: A dynamic Bayesian network from [21] and the dynamic programming approach from [22]. Table 1 shows resulting scores on both datasets.

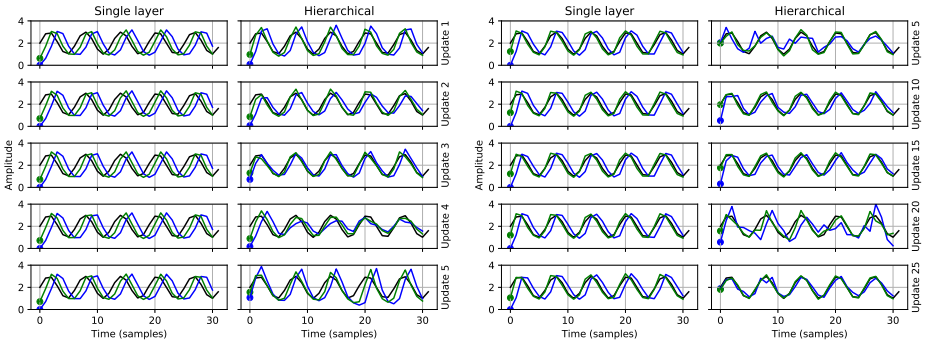
**Table 1.** Beat tracking evaluation.

| **SMC MIREX** | F-Score | CMLt | AMLt |
|---|---|---|---|
| Ellis [22] | 0.339 | 0.162 | 0.315 |
| Grosche [18] | 0.360 | 0.071 | 0.221 |
| Böck - online [21] | **0.521** | **0.363** | **0.433** |
| PCN (ours) | 0.205 | 0.108 | 0.201 |

| **NMED-T** | F-Score | CMLt | AMLt |
|---|---|---|---|
| Ellis [22] | 0.277 | **0.195** | **0.473** |
| Grosche [18] | 0.305 | 0.037 | 0.125 |
| Böck - online [21] | 0.092 | 0.105 | 0.280 |
| PCN (ours) | **0.321** | 0.111 | 0.295 |

Interestingly, with respect to the F-Measure, our method outperforms the baselines on the NMED-T dataset but delivers the worst performance on the SMC dataset. This indicates a useful performance on genres with salient rhythmical features, as the NMED-T dataset was designed focusing on Pop songs with clear rhythms. The SMC dataset features many songs with soft onsets, such as strings, where the novelty function from the prediction error is not sufficient. We hope that these encouraging results motivate future work with improved tracking based on predictive coding.

### 4.2 Audio Filtering with Top-Down Predictions




a) Repeated audio prediction with 10 state updates per timestep and 5 updates of the sequence prior.

b) Repeated audio prediction with 15 state updates per timestep and 25 updates of the sequence prior.

**Fig. 3.** a) Repeated prediction of a constant sine wave with single layer (left) and hierarchical PCN with two layers (right). The hierarchical model learns a top-down state prior for the sequence, while the single layer model has only local context. When convergence in the lowest layer is not guaranteed, such as with too few gradient descent steps or with inappropriate initialisation of precision, only the hierarchical model correctly tracks the incoming signal. b) With increased gradient steps for state inference in the lowest layer both single-layer and hierarchical PCN eventually show accurate posterior predictions (green). Predictions from the state prior (blue) improve only for the hierarchical model.

Figure 3 shows examples for repeated block-wise prediction of the same audio input with a single layer PCN and a hierarchical PCN with two layers for different gradient steps. In both networks, the inferred state and transition weights of the lowest layer are reset after each sequence prediction. This means that predictions in the single layer PCN are based on local information, i.e. the previously seen samples in the sequence. The hierarchical PCN keeps a top-down prediction of the lower layer's hidden state, providing refined contextual information for each prediction. This learnable state prior noticeably leads to a shifted starting point for the lowest layer in the hierarchical PCN in Fig. 3 a), where the lowest layer has not enough time to converge properly. When initialised with optimised parameters, both variants are able to approximate the target audio to a reasonable degree and the differences in prediction (and associated prediction errors) are largely restricted to the start of the sequence, as visible in Fig. 3 b). This indicates that minimizing prediction error can be solved through

online inference in independent trials as well as through the more gradual process of weights learning when information between trials is carried over. As noticeable in both Fig. 3 a) and b), the learning dynamic of the hierarchical model is significantly more dynamic, since the weighting of the top-down state prior is slightly adapted at each timestep.

The posterior predictions, indicated in Fig. 3 with green lines, show that the lowest PCN layer does not directly adapt to the top-down prior, but needs some time to tune the remaining weights to this additional source of information. When the top-down prior is correctly integrated, however, the hierarchical model quickly improves over the single layer model, especially with parameter initialisation that prevents full convergence of prediction errors in the lowest layer.

### 4.3 Replicating Filter Transfer Functions

We tested the possibility to simulate a Butterworth low-pass (LP) filter, which is widely in various DSP applications. Figure 4 shows input and output audio signals to the targeted LP filter and the corresponding in and outputs of a PCN. We test PCNs with single and two layers on a constantly ascending sine wave tone superimposed on constant white noise. Both PCN variants are able to replicate the desired transfer function of the LP filter and show the desired high frequency content removal.

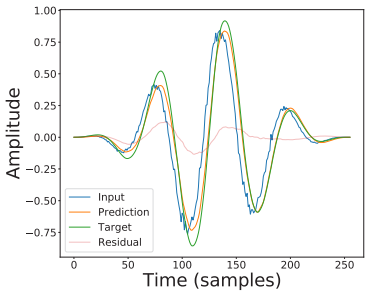



**Fig. 4.** Replicating an order 2 Butterworth LP filter. LP filter and PCN remove high frequency contents and have comparable output magnitudes. As the prediction starts with randomized states and without top-down prior, the prediction error (red) is higher at the sequence start.

## 5 Conclusion

We presented a gradient-based predictive coding model for audio analysis and synthesis. The hierarchical model targets biological plausibility through locally informed updates while still being efficient and accurate enough to replicate classical DSP tasks like filtering and beat tracking. We reviewed the similarities between the autoregressive state-space models underlying predictive coding, IIR filters, recurrent neural networks, and Kalman filtering. The model provides a basis for future work that could approach more complex DSP applications or subjectivity in artificial music perception.

## References


1. Kumar, S., Sedley, W., Nourski, K. V., Kawasaki, H., Oya, H., Patterson, R. D., Howard III, M. A., Friston, K. J., Griffiths, T. D.: Predictive coding and pitch processing in the auditory cortex. Journal of Cognitive Neuroscience, 23(10):3084–3094 (2011)
2. Friston, K., Kiebel, S.: Predictive coding under the free-energy principle. Philosophical Transactions of the Royal Society of London B: Biological Sciences, 364(1521):1211–1221 (2009)
3. Millidge, B., Tschantz, A., Buckley, C.: Predictive coding approximates backprop along arbitrary computation graphs. arXiv preprint arXiv:2006.04182 (2020)
4. Skerritt-Davis, B., Elhilali, M.: Computational framework for investigating predictive processing in auditory perception. Journal of Neuroscience Methods, 109177 (2021)
5. Miguel, M. A., Sigman, M., Fernandez Slezak, D.: From beat tracking to beat expectation: Cognitive-based beat tracking for capturing pulse clarity through time. PloS one, 15(11):e0242207 (2020)
6. Kuznetsov, B., Parker, J. D., Esqueda, F.: Differentiable IIR filters for machine learning applications. In: Proc. Int. Conf. Digital Audio Effects (eDAFx-20), pp. 297–303 (2020)
7. Adams, R. A., Friston, K. J., Bastos, A. M.: Active inference, predictive coding , cortical architecture. In: Recent Advances on the Modular Organization of the Cortex, 97–121. Springer (2015)
8. Marino, J.: Predictive coding, variational autoencoders, and biological connections. arXiv preprint arXiv:2011.07464 (2020)
9. Friston, K.: Hierarchical models in the brain. PLoS computational biology, 4(11), e1000211 (2008)
10. Millidge, B., Tschantz, A., Seth, A., Buckley, C.: Neural kalman filtering. arXiv preprint arXiv:2102.10021 (2021)
11. Irshad A., Salman M.: State-space approach to linear predictive coding of speech—a comparative assessment. In: IEEE 8th Conf. on Ind. Electronics and Applications (ICIEA), pp. 886–890. (2013)
12. Kondoz, A. M.: Digital speech: coding for low bit rate communication systems. John Wiley & Sons (2005)
13. O'Shaughnessy, D.: Linear predictive coding. IEEE potentials, 7(1):29–32 (1988)
14. Elman, J. L.: Finding structure in time. Cognitive science, 14(2):179–211 (1990)
15. Kalman, R. E.: A new approach to linear filtering and prediction problems. (1960)
16. Holzapfel, A., Davies, M. E., Zapata, J. R., Oliveira, J. L., Gouyon, F.: Selective sampling for beat tracking evaluation. IEEE Transactions on Audio, Speech, and Language Processing, 20(9):2539–2548 (2012)
17. Losorelli, S., Nguyen, D. T., Dmochowski, J. P., Kaneshiro, B.: NMED-T: A tempo-focused dataset of cortical and behavioral responses to naturalistic music. In: Proc. of the 18th Int. Society for Music Information Retrieval Conference (2017).
18. Grosche P., Muller, M.: Extracting predominant local pulse information from music recordings. IEEE Transactions on Audio, Speech, and Language Processing, 19(6):1688–1701 (2010)
19. Davies, M. E., Degara, N., Plumbley, M. D.: Evaluation methods for musical audio beat tracking algorithms. Queen Mary University of London, Centre for Digital Music, Tech. Rep. C4DM-TR-09-06, (2009)
20. Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., Ellis, D.: mir_eval: A transparent implementation of common mir metrics. In: Proc. of the 15th Int. Society for Music Information Retrieval Conference (2014)
21. Böck, S., Krebs, F., Widmer, G.: A multi-model approach to beat tracking considering heterogeneous music styles. In: Proc. of the 15th Int. Society for Music Information Retrieval Conference, (2014)
22. Ellis, D. P.: Beat tracking by dynamic programming. Journal of New Music Research, 36(1):51–60 (2007)
23. Schwartenbeck, P., Passecker, J., Hauser, T. U., FitzGerald, T. H., Kronbichler, M., Friston K. J.: Computational mechanisms of curiosity and goal-directed exploration. Elife, 8:e41703 (2019)