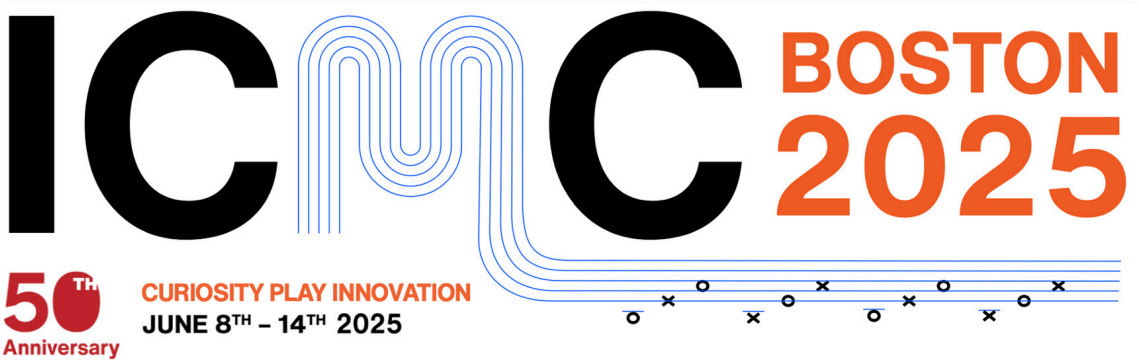

https://icmc2025.sites.northeastern.edu/

CURIOSITY • PLAY • INNOVATION

A 50th Anniversary Celebration of Creativity in Music, Science and Technology

BOSTON, MA USA I JUNE 8 – JUNE 14, 2025

Hosted by Northeastern University, New England Conservatory, Berklee College of Music, Boston Conservatory at Berklee and Emerson College, with special collaboration from the Massachusetts Institute of Technology (MIT) Music Department and the MIT Media Lab's Opera of the Future Group; the Central Conservatory of Music (Beijing); and IRCAM (Paris) – Institut de Recherche et Coordination Acoustique/Musique (Institute for Research and Coordination of Acoustic Music).

ICMA 2025 Paper Award

Each year the ICMA recognizes the best paper submitted with the Best Paper Award. The top scoring papers written by ICMA member are given toa panel elected by the ICMA Board, and a winner is decided from among these top submissions.

ICMA BEST PAPER AWARD

Resonate: Efficient Low Latency Spectral Analysis of Audio Signals

Alexandre R. J. François

The Best Paper award entails guaranteed publication of an extended version of the paper and an article in the Computer Music Journal.

2025 ICMC BEST PAPER AWARD COMMITTEE

Anthony Paul De Ritis
ICMA Research Coordinator
[Ex officio]

Victor Zappi
ICMC Boston 2025 Track Co-Chair

Akito van Troyer
ICMC Boston 2025 Track Co-Chair

Juan Parra
ICMA Board Member and Publications Coordinator

Douglas Keislar
Editor, Computer Music Journal

Tae Hong Park
ICMA Board Member

Henrik von Coler
ICMA Board Member

Resonate: Efficient Low Latency Spectral Analysis of Audio Signals

Alexandre R. J. François
Interactions Intelligence
alex@interactionsintelligence.org

ABSTRACT

This paper describes Resonate, an original low latency, low memory footprint, and low computational cost algorithm to evaluate perceptually relevant spectral information from audio signals. The fundamental building block is a resonator model that accumulates the signal contribution around its resonant frequency in the time domain, using the Exponentially Weighted Moving Average (EWMA). A compact, iterative formulation of the model affords computing an update at each signal input sample, requiring no buffering and involving only a handful of arithmetic operations. Consistently with on-line perceptual signal analysis, the EWMA gives more weight to recent input values, whereas the contributions of older values decay exponentially. A single parameter governs the dynamics of the system. Banks of such resonators, independently tuned to geometrically spaced resonant frequencies, compute an instantaneous, perceptually relevant estimate of the spectral content of an input signal in real-time. Both memory and per-sample computational complexity of such a bank are linear in the number of resonators, and independent of the number of input samples processed, or duration of processed signal. Furthermore, since the resonators are independent, there is no constraint on the tuning of their resonant frequencies or time constants, and all per sample computations can be parallelized across resonators. The cumulative computational cost for a given duration increases linearly with the number of input samples processed. The low latency afforded by Resonate opens the door to real-time music and speech applications that are out of the reach of FFT-based methods. The efficiency of the approach could reduce computational costs and inspire new designs for low-level audio processing layers in machine learning systems.

1. INTRODUCTION

This paper describes *Resonate*, an original low-latency, low memory footprint, and low computational cost algorithm to evaluate perceptually relevant spectral information from audio signals. *Resonate* is particularly relevant in the context of real-time, perceptually motivated signal analysis. It could also reduce computational costs and guide new designs for low-level audio processing layers in machine learning systems.

The next section provides context and motivation for a different approach to spectral analysis of audio signals. Section 3 describes the details of *Resonate*'s fundamental building block, a resonator model that accumulates the signal contribution around its resonant frequency in the time domain, using the Exponentially Weighted Moving Average. Section 4 demonstrates the use of banks of such resonators, independently tuned to geometrically spaced resonant frequencies, to compute an instantaneous, perceptually relevant estimate of the spectral content of an input signal. The paper concludes with a summary of contributions and a list of online resources made available to encourage adoption in the wider community.

2. CONTEXT AND MOTIVATION

Most music and speech analysis applications count as a major feature the frequency spectrum of the input signal as a function of time, often represented as a spectrogram. This information is typically obtained by computing the Short Time Fourier Transform (STFT) over the signal, i.e. a sequence of Fast Fourier Transforms (FFTs) of windowed data frames, where the window slides forward through time, usually with some overlap and shaping to avoid introducing discontinuities that would cause high-frequency artifacts. Established signal processing textbooks document in detail the many variants and their properties of this approach, rooted in the FFT, see e.g. [1][2][3].

In the context of audio processing, the input is a real-valued signal $x(t) \in [-1, 1]$, regularly sampled at sampling rate sr. The FFT algorithm efficiently computes the Discrete Fourier Transform (DFT) of a portion of the signal, that is the spectrum value $X(\omega_k)$ of input signal x for each frequency band ω in the appropriate discrete truncated Fourier expansion. Equation 1 shows the mathematical expression of the n^{th} term of the DFT over a window of N samples, where $x(t_n)$ is the amplitude of the input signal at time t_n. The power (squared magnitude) or magnitude of the complex number $X(\omega_k)$ captures the strength of the term's frequency in the input signal. Note that Equation 1 describes a dynamic system that oscillates with greater amplitude at frequency $f_k = \frac{\omega_k}{2\pi}$, namely a *resonator* of resonant frequency f_k.

$$X(\omega_k) = \sum_{n=0}^{N-1} x(t_n) e^{-i\omega_k t_n} \quad (1)$$

Despite its legendary efficiency and the ubiquitous availability of hardware-accelerated implementations, the FFT comes with constraints that make it less than ideal for perceptually motivated and real-time audio processing applications.

First, computing an FFT requires buffering the input data. In the context of real-time applications, this imposes an inherent lower bound on the minimal amount of delay introduced by analysis computations. In the FFT computation, there is an inherent trade-off between frequency resolution and time precision, determined by the number N of samples in the buffer and the signal's sampling rate sr. The minimum delay, without accounting for any computation, is the buffer duration $d = N/sr$. At audio frame sizes of 128, 256 or 512 samples, at a sampling rate of 44100Hz, the minimum theoretical corresponding delays amount to approximately 3ms, 6ms or 12ms respectively. Human auditory perception is sensitive to a few milliseconds of latency [4]; in studio settings, musicians find latencies above 10-12ms distracting. Real-time audio-to-MIDI software such as Jam Origin's MIDI guitar [5] or Vochlea's Dubler [6], whose algorithms and models are proprietary, operate in constrained frequency range for specific instruments. The typical recommended latency setting is 128 samples or less (2.9ms at 44100Hz sampling), but lower latency comes at the cost of often prohibitively increased CPU load.

Second, the FFT's computational efficiency comes from a very clever and elegant economy of scale, which nonetheless introduces extra costs in memory, and trade offs in time and frequency accuracy. Specific applications often require further processing to extract domain-relevant information that is not directly computed by the FFT. By construction, the number of frequency bins in the FFT is equal to the number of samples in the input frame. Because all frequency bins are processed from the same signal frame, the *same* trade off in frequency and time resolution, dictated by sample rate and buffer length, applies to all frequencies. Bin frequencies are linearly distributed over the range defined by the sampling rate and the buffer size. However, humans perceive audio frequencies on a logarithmic scale [7]. For these reasons, music and speech applications often rearrange data from the linearly spaced frequency bins according to a mel-frequency scale [8] or a log-frequency scale. The standard in music applications, the Constant-Q Transform (CQT) [9] computes the signal's spectrum over geometrically spaced frequency bins, at different time scales in different parts of the spectrum. The CQT can be computed efficiently from FFTs [10][11], but the process results in increased memory and computation costs and additional delays, incompatible with real-time applications.

For example, Spotify's Basic Pitch [12][13] model, which performs audio-to-MIDI conversion with pitch bend detection, starts with a Constant-Q Transform of the input audio signal, which is down-sampled to keep memory and computational costs under control. Basic Pitch is not designed for (and is incompatible with) real-time inference.

To avoid these trade-offs, *Resonate* builds on a computationally efficient resonator model that does not require buffering. Although developed independently, *Resonate* shares some design principles and properties with the Sliding Windowed Infinite Fourier Transform (SWIFT) algorithm [14].

3. RESONATOR MODEL

Resonate's fundamental building block is a resonator model that accumulates the signal contribution around its resonant frequency in the time domain, using the Exponentially Weighted Moving Average (EWMA) [15], also known as a low-pass filter in signal processing. Consistently with on-line perceptual signal analysis, the EWMA gives more weight to recent input values, whereas the contributions of older values decay exponentially. A single parameter governs the dynamics of the system. The EWMA is computed online for each input sample, requiring no buffering and involving only a handful of arithmetic operations.

The resonator, characterized by its resonant frequency $f = \frac{\omega}{2\pi}$, is described by a complex number R whose amplitude captures the contribution of the input signal component around frequency f. Equation 2 describes the iterative update formulation for $R(\omega, t)$, where $\Delta t = 1/sr$ is the sample duration, and $\alpha \in [0, 1]$ is a constant parameter that dictates how much each new measurement affects the accumulated value. This formula computes a complex quantity that is analogous (but not equal) to $X(\omega)$ in Equation 1.

$$R(\omega, t) = (1 - \alpha)R(\omega, t - \Delta t) + \alpha x(t)e^{-i\omega t} \tag{2}$$

The term $e^{-i\omega t}$ in Equation 2 is a complex number whose amplitude, phase and angular frequency ω are fixed (this is known as a *phasor*), and whose values can be computed iteratively, as described in Equation 3.

$$e^{-i\omega t} = P(t) = P(t - \Delta t)e^{-i\omega \Delta t} \tag{3}$$

Substituting in Equation 2 yields a purely iterative expression for the resonator model, shown in Equation 4.

$$R(\omega_k, t) = (1-\alpha)R(\omega_k, t-\Delta t)+\alpha x(t)P(t-\Delta t)e^{-i\omega \Delta t} \tag{4}$$

Overall the algorithm for the resonator state update for each input sample x comes down to the rules shown in Equation 5.

$$\begin{aligned} P &\leftarrow Pe^{-i\omega \Delta t} \\ R &\leftarrow (1 - \alpha)R + \alpha xP \end{aligned} \tag{5}$$

The two complex numbers P and R capture the full state of the resonator. Updating the state at each input signal sample only requires a handful of arithmetic operations. Calculating the power and/or magnitude is not necessary for the update, and can be carried out only when required by the application, relatively efficiently as well.

The theoretical minimal delay introduced is limited by the input signal's sampling rate, as a value can be obtained for each sample. At a sampling rate of 44100 Hz, the sample duration is 22.7μs. The time added by the update and power or magnitude calculations on modern hardware will typically be of the same order as, or negligible with respect to sample duration. These numbers are several orders of magnitude smaller than those regularly encountered in FFT-based methods, and are compatible with real-time audio signal analysis-based music applications.

3.1 Dynamics

The dynamic system described by Equation 2 when the input signal x is a sinusoid of frequency f (resonator's resonant frequency) admits an attractor point R_f whose phase

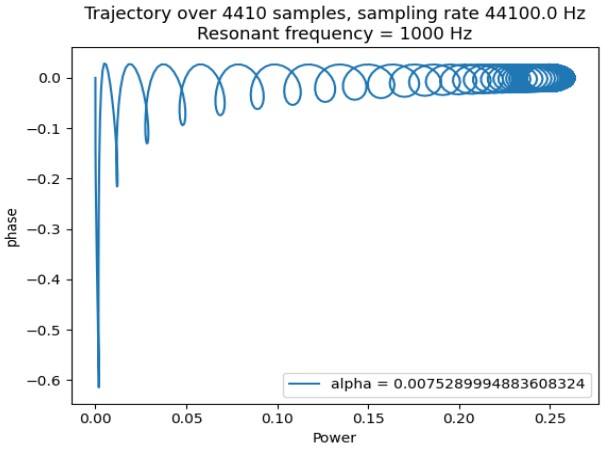

Figure 1. Trajectory in power-phase space of the response to a sinusoidal step signal tuned to the resonator's resonant frequency. The trajectory starts in the lower left corner and stabilizes around attractor point [0.25, 0.0]

is that between the resonator and the input signal's sinusoids, and whose magnitude is the integral of a squared sinusoid (from the product of the resonator's and the input signal's), that is, 0.5 under our assumptions, and 0.25 for the power (squared magnitude). Figure 1 shows the trajectory in power-phase space of the response to a sinusoidal step signal tuned to the resonator's resonant frequency.

The single parameter α dictates how much each new measurement affects the accumulated value, and therefore how quickly the resonator adjusts to a sustained change in the input signal (time resolution). In data analysis applications, this parameter is usually set according to domain- or data-dependent heuristics. As α regulates the dynamics of the system, it is useful to relate it to a time quantity to formulate a suitable heuristic.

By definition, the time constant τ of an exponential moving average is the amount of time for the smoothed response of a unit step function to reach $1 - e^{-1} \approx 62\%$ of the original signal. Equation 6 gives the relationship between this time constant and the smoothing factor α, with Δt the sampling time interval of the discrete time implementation.

$$\begin{aligned} \alpha &= 1 - e^{-\Delta t/\tau} \\ \tau &= -\frac{\Delta t}{\ln(1-\alpha)} \end{aligned} \tag{6}$$

In the *Resonate* model, the time constant controls both time and frequency resolution, and should therefore be a function of the resonator's resonant frequency. Intuitively, the time constant should be larger for lower frequencies as values need to be accumulated over a longer period of time to increase frequency resolution. Consequently, it seems reasonable to set the time constant to a multiple of the resonator's resonant cycle duration. Practically, the heuristic $\tau_f = \frac{log(1+f)}{f}$ yields reasonable results across resonant frequencies in the range of interest (see Figure 3 below). Substituting τ in 6 gives Equation 7.

$$\alpha_f = 1 - e^{-\Delta t \frac{f}{log(1+f)}} \tag{7}$$

As illustrated in Figure 2(a), the power of the resonator oscillates around the steady state value. Applying the EMWA with the same time constant α to the complex R from Equation 5, yields a stabilized value $\tilde{R}$ as shown in Equation 8. Figure 2(b) shows the smoothed power response over time of three resonators tuned at resonant frequencies of different orders of magnitude, to a sinusoidal step signal tuned to the resonator's resonant frequency. Figure 2(c) shows the smoothed steady state power response of three resonators tuned at resonant frequencies of different orders of magnitude, to a step sinusoidal signal tuned at various frequencies across the range of interest.

$$\tilde{R} \leftarrow (1-\alpha)\tilde{R} + \alpha R \tag{8}$$

Informal experiments suggest that the model is robust with respect to specific values of α: the behavior of the resonator remains consistent around values with the same order of magnitude.

3.2 Phase

When the input signal's frequency is close to the resonator's resonant frequency, the system oscillates around the attractor point R_f with heightened but non-maximal average power. In this mode, the phase between input signal and resonator carries significant information. The computed phase between the input component and that of the resonator constantly shifts at a rate proportional to the difference in wavelength between the signal's frequency and the resonant frequency. This is because the instantaneous interpretation of a slightly shorter or longer wavelength in the input signal is identical to an instantaneous phase shift (aperture effect). Equation 9 expresses the actual signal frequency f' from the phase shift $\Delta\phi$ over duration ΔT.

$$f' = f - \frac{\Delta\phi}{2\pi\Delta T} \tag{9}$$

Possible applications include a tuning systems which provides feedback to a user attempting to manually tune an instrument to match the resonant frequency of a resonator, and an adaptive resonator that can track the possibly evolving frequency of a simple input signal, provided the initial tuning is close enough to the starting signal frequency.

Assuming a signal source of known fixed frequency f, the Doppler shift in frequency f' measured at an observer (microphone) gives the relative velocity v of the source and the observer. Equation 10 gives the mathematical formula for v, as a function of f, f' and the speed of sound c.

$$v = c\frac{f' - f}{f} \tag{10}$$

Audio signal spectral analysis applications require more than a single resonator, as explored in the next section.

4. SPECTRAL ANALYSIS

4.1 Resonator bank

The resonator model introduced above is particularly suitable for building banks of independently tuned resonators, similar in principle to banks of non-linear band-pass filters.

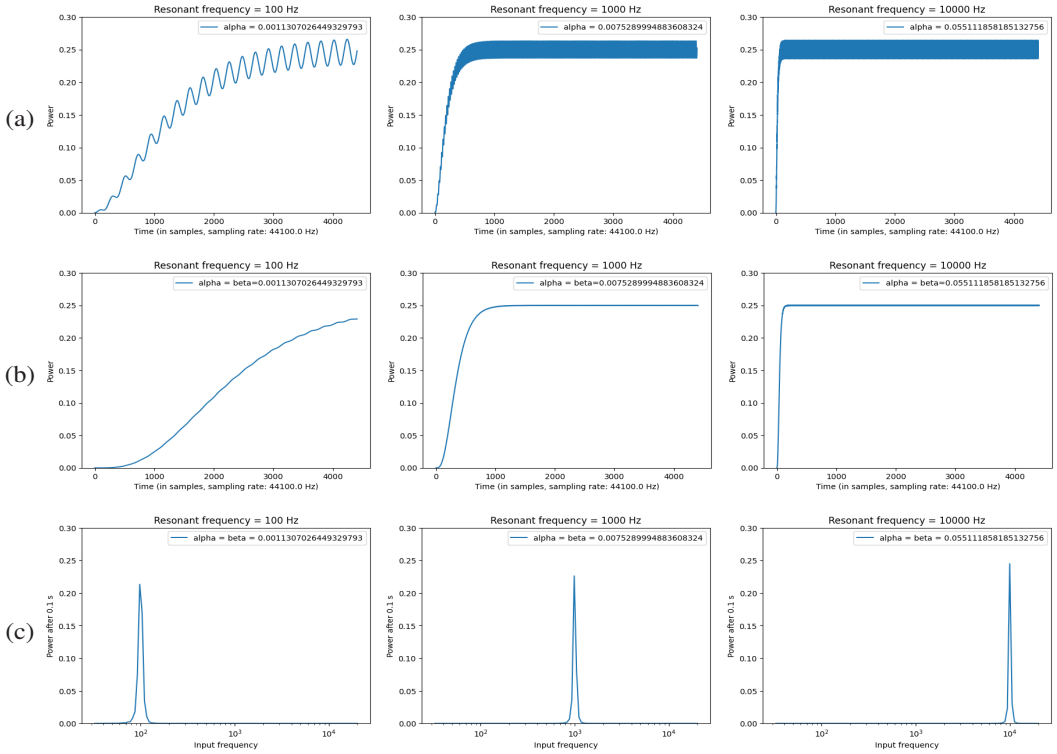

Figure 2. (a) Power response over time to a sinusoidal step signal tuned to the resonator's resonant frequency. (b) Smoothed power response over time to a sinusoidal step signal tuned to the resonator's resonant frequency. (c) Smoothed steady state power response to a step sinusoidal signal tuned at various frequencies across the range of interest. Sampling rate: 44100Hz

The resonators are independent, therefore there is no external constraint on the tuning of their resonant frequencies or time constants. Both memory and per-sample update computational complexity of such a bank are linear in the number of resonators, and independent of the number of input samples processed or duration of processed signal (or portion thereof). Because update computations for each resonator are independent, per-sample computations can be parallelized to further reduce computation time and therefore reduce latency, for example by taking advantage of Single Instruction Multiple Data (SIMD) architectures where available. The cumulative computational cost for a given signal duration increases linearly with the number of input samples processed.

In the context of audio processing, the frequency range of interest is typically 20-20000Hz. Choosing a resonator bank makeup that emulates the geometrically spaced frequency bins of the CQT yields results directly relevant to human auditory perception and provides grounds for comparison with an established spectral analysis algorithm.

Figure 3 shows the response (power at steady state) from a bank of 112 resonators tuned to geometrically spaced frequencies, to sinusoidal step signals of various frequencies spanning the range of interest. The graph shows that the heuristic established for the resonators' time constants produces a reasonably regular and uniform coverage of the frequency range. Note that the resonator's time constants could be learned, or at least tuned, to better reflect a specific data corpus and/or capture pipeline (e.g. microphone response across the frequency range).

The graph suggests a large amount of overlap in the response of the 112 resonators in the bank. Such redundancy could make for a system that is highly robust to decimation. Some applications, on the other hand, may not require such a tight coverage of the frequency range of interest, leading to further reduction in memory and computational costs.

4.2 Spectrogram

Spectral information as a function of time is typically presented graphically in the form of a spectrogram. Figure 4 shows the spectrograms of a short musical excerpt (electric piano) of duration 11.15 s (491904 samples at 44100Hz), computed from the CQT and from *Resonate*. Both spectrograms were plotted in a Python environment using Librosa's [16] `specshow` function. The CQT implementation, also from Librosa, is based on [11]. In the spectrograms, the vertical axis represents frequency, so that each row of the image corresponds to a frequency bin for the CQT and to a resonator for *Resonate*. To facilitate comparison, the number of resonators is the same as the number of CQT frequency bins and the resonator's resonant frequencies correspond to the bin frequencies. The horizontal axis represents time. For the CQT, the hop length determines the duration of each slice of time (image column), in this case 512 samples (about 11.6ms at 44100Hz). The color value at each pixel represents the power (in decibels) in the input signal at the corresponding frequency over the time slice, relative to the maximum power over the whole

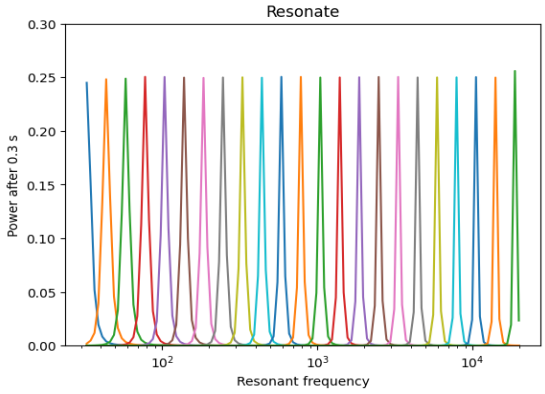

Figure 3. Response from a tuned resonator bank to sinusoidal step signals of various frequencies across the range of interest. The bank comprises 112 resonators tuned to geometrically spaced frequencies from 32.7Hz to 19910.18Hz, with 12 resonators per octave. The input signal sampling rate is 44100Hz.

excerpt. The color map used here assigns brighter colors to higher power. The *Resonate* spectrogram similarly plots the power (in decibels) computed from the resonators. For direct comparison with the CQT-based plot, here each column of the spectrogram shows the state of the resonator bank taken every 512 samples. As expected, the images are not identical, yet seem to capture similar features of the input signal.

On an Apple Mac mini M1 (2020), the CQT computation for this musical segment takes approximately 0.05s to 0.1s. The iterative nature of the *Resonate* algorithm makes it difficult to vectorize across samples. A Python implementation that loops over the samples to compute the spectrogram takes approximately 1.7s. A C++ implementation that loops over the samples more efficiently and leverages the Accelerate framework to vectorize updates per sample across resonators, called from Python under the same conditions, takes on the order of 0.05s. Table 1 shows CQT and *Resonate* computation times with various hop lengths. CQT computation times increase significantly as hop length decreases, while the corresponding *Resonate* computation times remain essentially constant. As already pointed out, *Resonate*'s memory footprint is much lighter than the CQT's, and only grows linearly with the number of oscillators. Naturally, when computed and stored offline, the output size grows linearly with the number of time slices (inversely proportional to the hop length).

Resonate computes spectrogram values at the input sampling rate with delays easily compatible with real-time applications on modern hardware, with a time resolution that is out of reach of FFT-based methods such as the CQT. Figures 5, 6 and 7 illustrate the level of detail afforded by *Resonate* spectrograms. Figure 5 shows spectrogram details of solo trumpet music, revealing spectral characteristics of the instrument at a high temporal resolution. Figure 6 shows spectrogram details of rock music excerpt with strong, well defined rhythmic patterns clearly visible. Figure 7 shows spectrogram details of a voice recording (short reading excerpt). *Resonate* spectrogram data could provide richer information to AI models at reduced memory and computational costs. The efficiency of the approach

hop length	Computation time (s)	
	CQT	Resonate
512	0.05	0.05
256	0.06	0.05
128	0.08	0.05
64	0.2	0.05
32	0.5	0.05
16	1.2	0.05
8	4.5	0.05
4	80	0.07
2	*	0.08
1	*	0.1

Table 1. Approximate running times (in s) for CQT and *Resonate* computations for the same short musical excerpt as in Figure 4 (491904 samples at 44100Hz), with various hop lengths.

could also inspire new designs for deep learning models suitable for inference from time sampled data in interactive use cases, where latency is critical and only present and past data are available.

The Oscillators app [17] demonstrates real-time *Resonate* spectrograms and *Resonate*-powered audio features in an interactive setting. The Resonate Youtube playlist [18] features video captures of real-time demonstrations.

4.3 Inverse and synthesis

The *Resonate* model is invertible, up to numerical approximations. Inverting Equation 2 (and the additional smoothing step), the successive sample values of the original signal can be recovered, at the original sampling rate, from the successive states of *any* single resonator, as shown in Equation 11.

$$R(\omega,t) = \frac{\tilde{R}(\omega,t) - (1-\alpha)\tilde{R}(\omega_k, t-\Delta t)}{\alpha}$$
$$x(t) = Re(\frac{(R(\omega,t) - (1-\alpha)R(\omega_k, t-\Delta t))}{\alpha} e^{i\omega t}) \quad (11)$$

From a synthesis perspective, since *Resonate* captures spectral and phase information, the successive states of a bank of resonators (or a subset thereof) can be used to generate an audio signal at the same sampling rate as the original's, in a process akin to additive synthesis. Equation 12 shows a possible formula for a synthesized signal, up to a proportionality constant K that depends on the set of resonators' coverage of the frequency range.

$$r(t) = KRe(\sum_{\omega} \tilde{R}(\omega,t)e^{i\omega t}) \quad (12)$$

This approach does not revert the effects of the EWMAs, so comparing the resulting signal and the original signal shows both temporal and numerical discrepancies. The signals should however exhibit similar patterns. Preliminary informal experiments on musical excerpts suggest that the signal synthesized from a resonator bank with appropriate frequency range coverage exhibits audio features similar to those of the original input signal. Experiments also suggest that a much smaller number of oscillators distributed across the frequency range of interest is sufficient

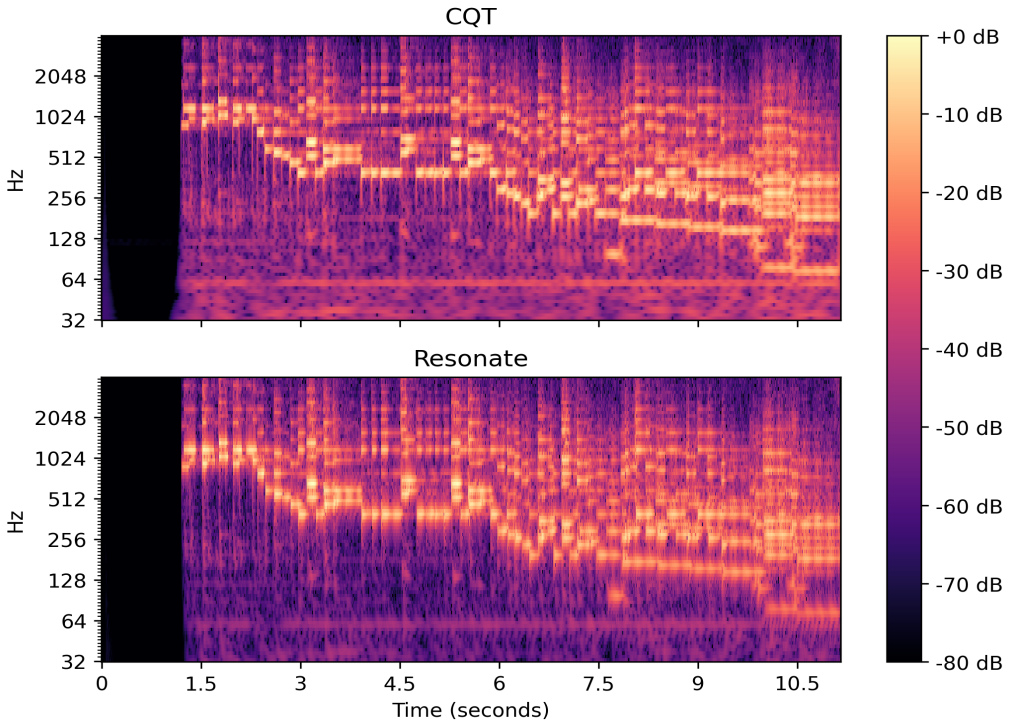

Figure 4. Spectrogram of a short musical excerpt (first 11.15s of *Who's Loving You* by The Jackson 5; electric piano) computed from the constant-Q transform (CQT) and from a Resonate implementation (spectrogram display and CQT from Librosa, sampling rate: 44100 Hz, hop length: 512 samples, 84 frequency bins from 32.7 Hz to 3950.7 Hz).

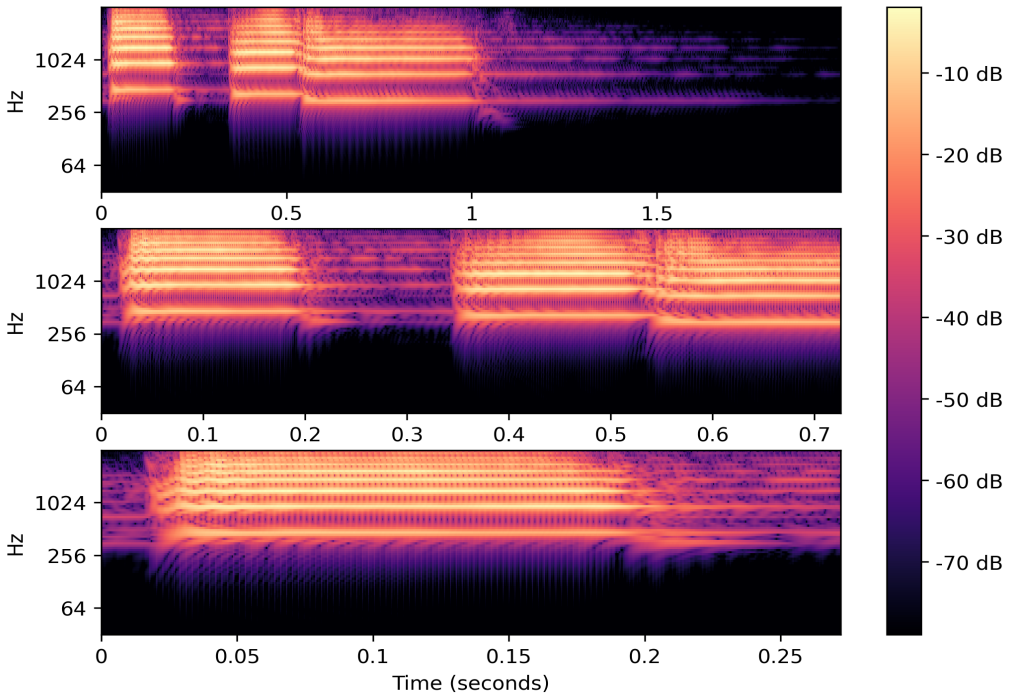

Figure 5. *Resonate* spectrogram details from Librosa's trumpet sample. Sampling rate 22050Hz.

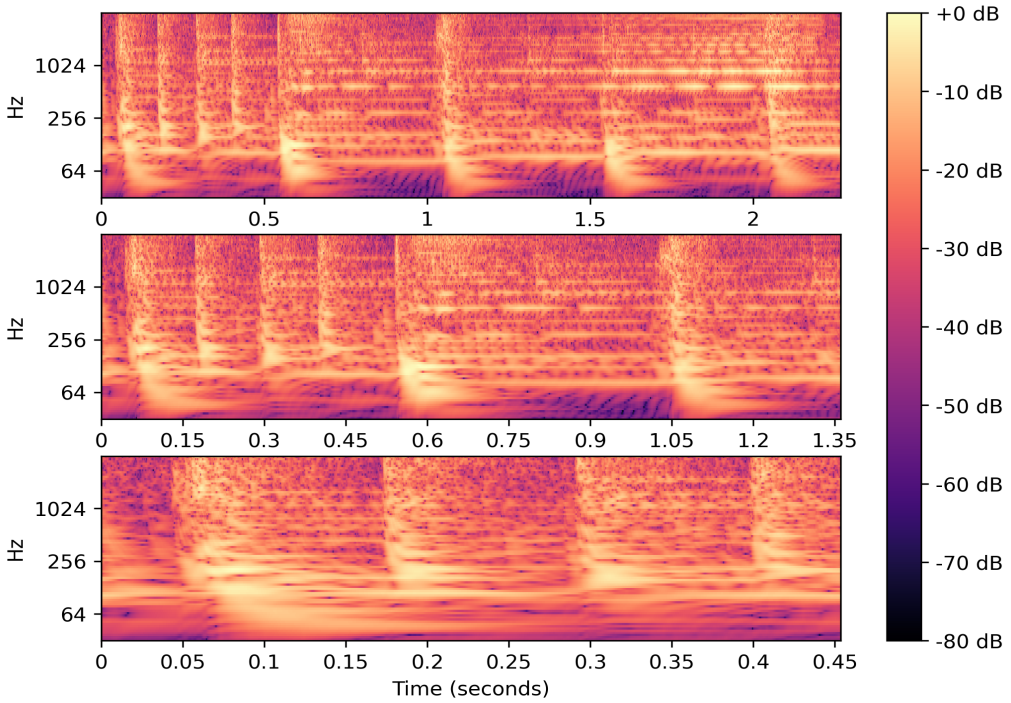

Figure 6. *Resonate* spectrogram details of a short excerpt from the Peter Gunn Theme by The Blues Brothers. Sampling rate 41000Hz.

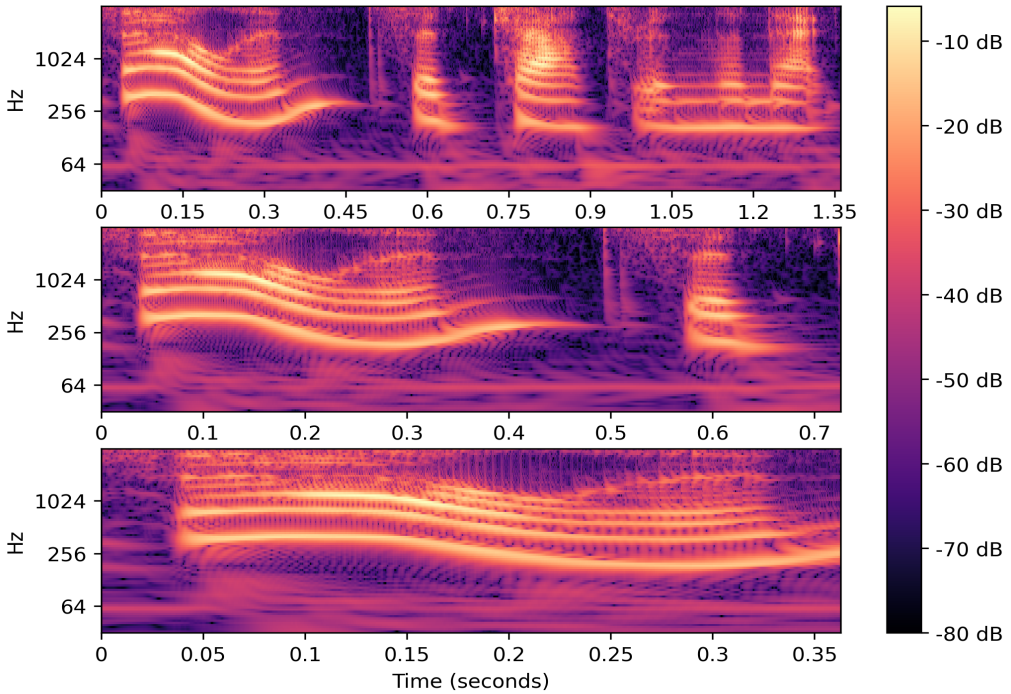

Figure 7. *Resonate* spectrogram details from Librosa's *libri3* voice recording sample (reading excerpt). Sampling rate 22050Hz.

to capture salient features of music signal. Such robustness to decimation of resonators seems consistent with that of human perception. It could also point to further potential savings in memory and compute by using banks with a smaller number of resonators in the low-level audio processing layers of deep learning systems, at least for some classes of applications.

5. SUMMARY AND FUTURE WORK

This paper described *Resonate*, an original low latency, low memory footprint, and low computational cost algorithm to evaluate perceptually relevant spectral information from audio signals. The model builds on a resonator model that accumulates the signal contribution around its resonant frequency in the time domain using the Exponentially Weighted Moving Average. A compact, iterative formulation of the model affords computing an update at each signal input sample, requiring no buffering and involving only a handful of arithmetic operations. Banks of such resonators, independently tuned to geometrically spaced resonant frequencies, compute an instantaneous, perceptually relevant estimate of the spectral content of an input signal in real-time. Both memory and per-sample computational complexity of such a bank are linear in the number of resonators, and independent of the number of input samples processed, or duration of processed signal. Furthermore, since the resonators are independent, there is no constraint on the tuning of their resonant frequencies or time constants, and all *per sample* computations can be parallelized across resonators. The cumulative computational cost for a given duration increases linearly with the number of input samples processed.

Resonate is particularly relevant in the context of real-time, perceptually motivated signal analysis, where its computational efficiency and low latency open the door to real-time music and speech analysis applications impossible with FFT-based methods. Furthermore, the efficiency of the approach could reduce computational costs and guide new designs for low-level audio processing layers in machine learning systems.

This paper described the principles of *Resonate*. Much work lies ahead to refine and validate the model and explore its potential use in existing and new applications. To encourage adoption in the wider community, the open source module `noFFT` [19] provides python and C++ implementations of *Resonate* functions and Jupyter notebooks with the code used to generate the figures in the paper. The Swift package `Oscillators` [20] offers reference implementations in Swift and C++.

6. REFERENCES

[1] J. O. Smith, *Spectral Audio Signal Processing*. ccrma.stanford.edu/j̃os/sasp, 2011.

[2] M. Vetterli, J. Kovacevic, and V. K. Goyal, *Foundations of Signal Processing*. www.fourierandwavelets.org, 2014.

[3] J. Kovacevic, V. K. Goyal, and M. Vetterli, *Fourier and Wavelet Signal Processing*. www.fourierandwavelets.org, 2013.

[4] H. Wallach, E. B. Newman, and M. R. Rosenzweig, "The Precedence Effect in Sound Localization," *The American Journal of Psychology*, vol. 62, no. 3, pp. 315–336, 1949. [Online]. Available: www.jstor.org/stable/1418275

[5] "MIDI Guitar." [Online]. Available: www.jamorigin.com

[6] "Dubler." [Online]. Available: vochlea.com

[7] R. Rasch and R. Plomp, "1 - The Perception of Musical Tones," in *Psychology of Music*, ser. Cognition and Perception, D. Deutsch, Ed. New York: Academic Press, 1982, pp. 1–24.

[8] S. Stevens and J. Volkmann, "The Relation of Pitch to Frequency; A Revised Scale," *The American Journal of Psychology*, vol. 53, pp. 329–353, 01 1940.

[9] J. C. Brown, "Calculation of a Constant Q Spectral Transform," *Journal of the Acoustical Society of America*, vol. 89, pp. 425–434, 01 1991.

[10] J. C. Brown and M. S. Puckette, ""An efficient algorithm for the calculation of a constant Q transform"," *Journal of the Acoustical Society of America*, vol. 92, pp. 2698–2701, 11 1992.

[11] C. Schörkhuber and A. Klapuri, "Constant-Q transform toolbox for music processing," *Proc. 7th Sound and Music Computing Conf.*, 01 2010.

[12] R. M. Bittner, J. J. Bosch, D. Rubinstein, G. Meseguer-Brocal, and S. Ewert, "Basic Pitch," 2024. [Online]. Available: github.com/spotify/basic-pitch

[13] ——, "A Lightweight Instrument-Agnostic Model for Polyphonic Note Transcription and Multipitch Estimation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Singapore, 2022.

[14] L. L. Grado, M. D. Johnson, and T. I. Netoff, "The Sliding Windowed Infinite Fourier Transform [Tips & Tricks]," *IEEE Signal Processing Magazine*, vol. 34, no. 5, pp. 183–188, 2017.

[15] *NIST/SEMATECH e-Handbook of Statistical Methods, 6.4.3.1. Single Exponential Smoothing*. NIST, retrieved January 2025.

[16] B. McFee *et al.*, "librosa/librosa: 0.10.2.post1," May 2024. [Online]. Available: doi.org/10.5281/zenodo.11192913

[17] A. R. François, "Oscillators App (iOS/iPadOS/MacOS)," 2025. [Online]. Available: www.alexandrefrancois.org/Oscillators

[18] ——, "Resonate Playlist," YouTube, 2025. [Online]. Available: www.youtube.com/playlist?list=PLVcB_ABiKC_cbemxXUUJXHAQsHEHxPOP1

[19] ——, "noFFT," 2025. [Online]. Available: github.com/alexandrefrancois/noFFT

[20] ——, "Oscillators (Swift Package)," 2025. [Online]. Available: github.com/alexandrefrancois/Oscillators