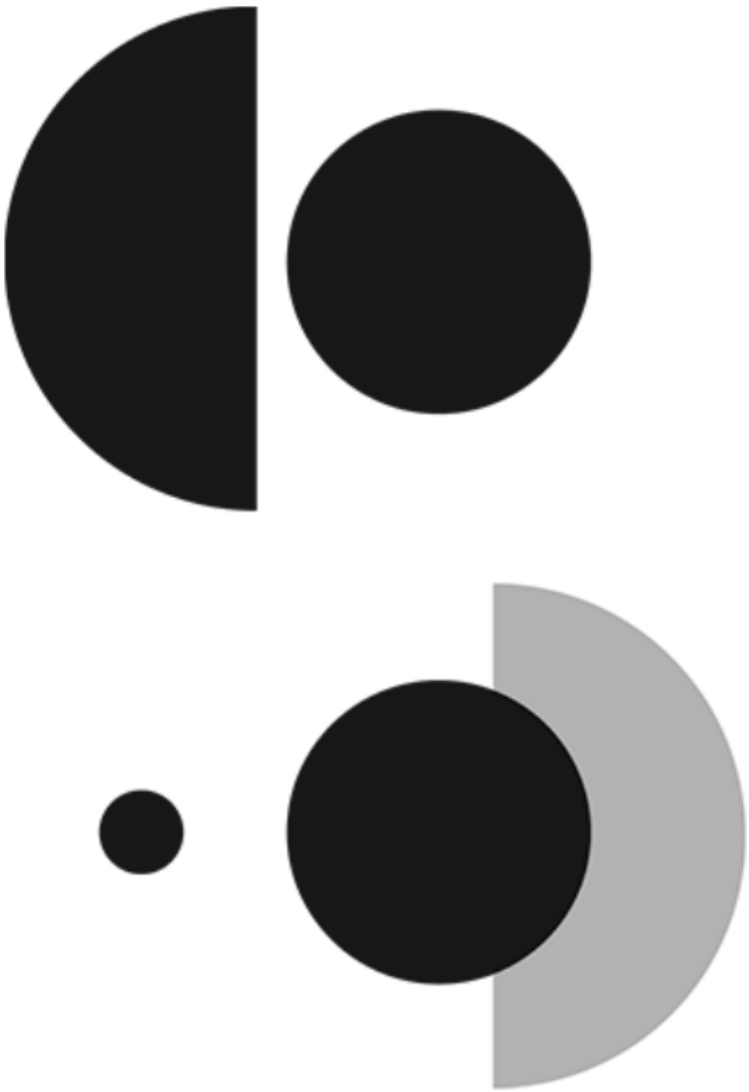

# International Computer Music Conference 2026

## May 10-16, 2026

## Hamburg, Germany

# Real-Time, Low-Latency, High Resolution Audio Spectral Analysis: Phase Matters

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

## ABSTRACT

*This paper introduces an original approach to computing a spectral representation of audio signals, with high temporal and frequency resolution and high amplitude accuracy, in real-time and with low latency. Applying techniques from phase vocoders to make use of phase information, a new tracking resonator model extends the original Resonate model while retaining its iterative formulation and computational efficiency. A bank composed of frequency tracking resonators constantly self-tunes to the contents of the input signal, rendering the precise tuning of the resonators irrelevant, as long as the bank offers an appropriate coverage of the frequency range of interest for the target application. Self-tuning banks form the basis for an analysis technique that produces, in real-time, for each input sample, a list of uniquely identified and precisely tracked frequency components present in the input signal, together with their correct amplitudes. High temporal and frequency resolution spectrograms illustrate the spectral analysis of real musical signals in a familiar format. The detailed representations produced can potentially improve the quality and accuracy of any traditional application. They also offer promising prospects for real-time, low-latency applications such as accompaniment and improvisation systems. Encouraging initial synthesis experiments also motivate further investigation.*

## 1. INTRODUCTION

This paper introduces an original approach to computing a spectral representation of audio signals with high temporal and frequency resolutions and high amplitude accuracy, in real-time and with low latency. The method applies phase vocoder techniques to extend the *Resonate* [1] model. The focus remains on harmonic analysis with the ultimate goal of applicability to real-time, low latency scenarios. The paper is organized as follows. Section 2 gives a quick overview of the uses of phase vocoder principles that inspired this work. Section 3 summarizes the *Resonate* model in the context of instantaneous frequency computation, first in a single resonator, then in a resonator bank. Section 4 extends this model to a frequency tracking resonator and its application in self-tuning resonator banks. Such banks form the basis for an analysis technique that produces, in real-time, for each input sample, a list of uniquely identified and precisely tracked frequency components present in the input signal, together with their correct amplitudes. High temporal and frequency resolution spectrograms illustrate the spectral analysis of real musical signals in a familiar format. Section 5 presents some initial synthesis results. The paper concludes with a summary of contributions and directions for future research and applications.



## 2. PHASE MATTERS

Spectral analysis for audio applications typically focuses on amplitude or power spectrum plots or spectrograms, treating the phase data as a computational by-product. Yet the phase carries important information, as evidenced in the phase vocoder literature.

The phase vocoder is primarily used for time-scale modifications and pitch transpositions of audio signals. As an analysis-synthesis technique, the phase vocoder produces an output signal that is identical or similar to the input signal, by way of a representation that is suitable for meaningful time-frequency manipulations. Portnoff [2] laid the foundations for its practical use in digital signal processing with an implementation based on the Fast Fourier Transform (FFT). Moorer [3] and Dolson [4] explored its use for musical applications.

Phase information is crucial in both analysis and synthesis. The analysis stage computes, from the input signal, a spectral representation consisting of the time varying amplitudes and instantaneous frequencies, by estimating phase time differentials (temporal phase consistency). The synthesis stage makes use of the inverse Fourier Transform (iFFT) to produce an audio signal similar to the input signal from the possibly modified frequency domain representation produced in the analysis stage. If no modification occured, because of the properties of the FFT and iFFT, the output signal is mathematically identical to the input signal (i.e. the phase vocoder is an identity system). In case of modifications, phase constraints must be enforced in order to generate a high quality output signal.

The analysis phase of the FFT-based formulation actually applies the Short-Time Fourier Transform (STFT) to the input signal to produce a frame-based frequency domain representation. This comes at the cost of added mathematical complexity, which compounds in the synthesis phase [5]. The STFT has the advantage of being computationally efficient, but at the cost of significant constraints. For example, the frequency bands are fixed, distributed on

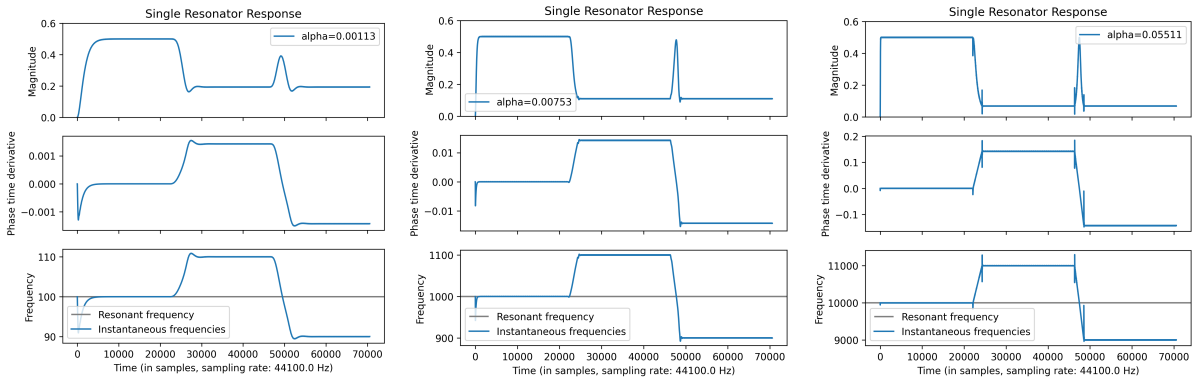


**Figure 1**. Magnitude, phase difference and estimated instantaneous frequencies over time in response to a sinusoidal signal of fixed amplitude 1, whose frequency varies around the resonator's resonant frequency: 100-110-90 Hz, 1000-1100-900 Hz and 10000-11000-9000 Hz; 0.05s transition duration.

a linear scale (while humans perceive frequencies on a logarithmic scale) and dictated by the window size and input signal sampling rate. Furthermore, the sliding window approach of the SFFT introduces ambiguity in the phase estimation, which necessitates extra processing (phase unwrapping). These and other characteristics inherent to the FFT pose challenges that affect computational complexity, quality of the analysis and of the synthesized output signal. Audio signals produced by phase vocoders exhibit characteristic features such as "transient smearing" and a number of effects referred to as "phasiness." Various analyses and mitigation techniques, in the context of musical applications, are the subject of a substantial body of work (see e.g. [6, 7, 8, 9]).

In contrast to the FFT, *Resonate* builds on 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 computes a magnitude and phase update at each input signal sample, requiring no buffering and involving only a handful of arithmetic operations. Banks of resonators, independently tuned to perceptually relevant frequency scales, 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. Since the resonators are independent, their resonant frequencies or time constants can be set arbitrarily, and all per sample computations can be parallelized across resonators.

These properties suggest a much simpler, efficient and possibly novel application of phase vocoder techniques, the first results of which are the subject of this paper.

## 3. INSTANTANEOUS FREQUENCIES

This section summarizes the *Resonate* model originally described in [1], in the context of instantaneous frequency computation, first in a single resonator, then in a bank of independently tuned resonators.

### 3.1 Single Resonator

Each *Resonate* resonator (indexed $k$), characterized by its resonant frequency $f_k = \frac{\omega_k}{2\pi}$, is described by a complex number $R_k$ whose amplitude captures the contribution of the input signal component around frequency $f_k$. Equation 2 shows the recursive update formula for $R_k$ by way of a phasor $P_k$, applied for each sample $x$ of a real-valued input signal $x(t) \in [-1, 1]$, regularly sampled at sampling rate $sr$. $\Delta t = 1/sr$ is the sample duration.

$$P_k(t) = P_k(t - \Delta t)e^{-i\omega_k \Delta t} \quad (1)$$

$$R_k(t) = (1 - \alpha_k)R_k(t - \Delta t) + \alpha_k x(t)P_k(t) \quad (2)$$

The parameter $\alpha_k \in [0, 1]$ dictates how much each new measurement affects the accumulated value; it can be expressed as a function of the system's time constant, set heuristically as a function of the resonant frequency $f$ (intuitively inversely proportional to the frequency). The formula in Equation 3 is a reasonable heuristic for the frequency range of interest in audio applications (20-20000 Hz).

$$\alpha_k = 1 - e^{-\Delta t \frac{f_k}{log(1+f_k)}} \quad (3)$$

A smoothed state $\tilde{R}_k$ is produced by applying the EMWA to $R_k$ with parameter $\beta_k$ to dampen power and phase oscillations, as shown in Equation 4. In practice, $\beta_k = \alpha_k$ yields satisfactory results.

$$\tilde{R}_k(t) = (1 - \beta_k)\tilde{R}_k(t - \Delta t) + \beta_k R_k(t) \quad (4)$$

This formulation is consistent with the first steps in the filterbank interpretation of the phase vocoder analysis as described by Dolson in [4], namely heterodyning followed by lowpass filtering. The amplitude at time $t$ of a *sinusoidal input signal of same frequency as the resonant frequency* is directly proportional to the magnitude of the complex number $\tilde{R}_k(t)$.

At each time step, the phase difference $\Delta\phi(t)$ between the current and previous value of $\tilde{R}$ provides an estimate of the phase's time derivative, to compute the corresponding instantaneous frequency, as shown in Equation 5.

$$f_k(t) = f_k(t - \Delta t) + \frac{\Delta\phi_k(t)}{2\pi\Delta T} \quad (5)$$

Using the property that the phase of a complex number multiplied by the conjugate of another complex number is the phase difference between the two numbers yields Equation 7 for $\Delta\phi_k(t)$, which involves only one principal value

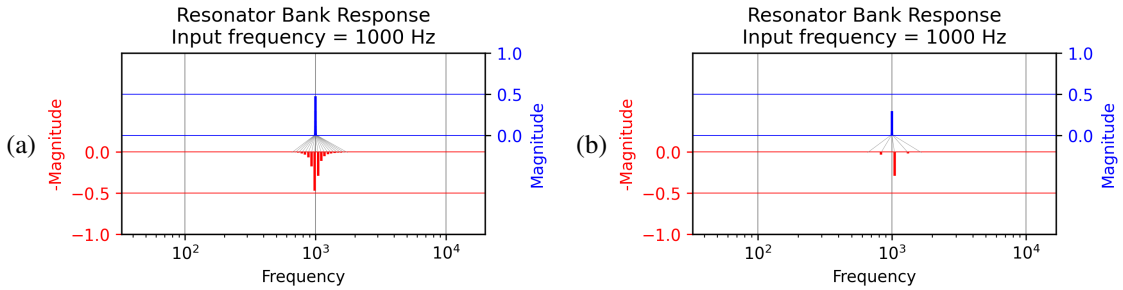


**Figure 2**. Steady state magnitudes for banks of resonators tuned at geometrically spaced frequencies, in response to a sinusoidal signal of fixed amplitude (1) and frequency (1000 Hz). The bottom graph plots the opposite of each resonator's magnitude at the resonator's resonant frequency position on the x-axis; the top graph plots the resonators' magnitudes at the instantaneous frequency. (a) 112 frequencies from 32.7 to 19910.2 Hz, (b) 28 frequencies from 32.7 to 16742.4 Hz. Sampling rate: 44100 Hz.

argument computation. Because $\Delta\phi_k(t)$ is computed for each input sample, there is no concern of phase unwrapping, a crucial necessary step in FFT-based vocoder analysis when the hop length is greater than 1 (or when explicitly computing and subtracting phases).

$$D_k(t) = \tilde{R}_k(t)\overline{\tilde{R}_k}(t - \Delta t) \quad (6)$$

$$\Delta\phi_k(t) = Arg(D_k(t)) \quad (7)$$

Finally, applying the EMWA with time constant $\gamma_k$ to the estimated frequency yields the stabilized estimate of Equation 8. In practice, $\gamma_k = \alpha_k/2$ yields satisfactory results.

$$\tilde{f}_k(t) = f_k(t - \Delta t) + \gamma_k \frac{\Delta\phi_k(t)}{2\pi\Delta T} \quad (8)$$

The complex numbers $P$, $R$ and $\tilde{R}$ capture the full state of the resonator. Updating the state at each input signal sample only requires a handful of arithmetic operations. Equations 6 and 8 add a few more for instantaneous frequency estimation. The argument computation of Equation 7, typically obtained with the `arctan2` transcendental function, is the only source of significant additional computational complexity and potential source of significant increase in processing time. Fortunately this operation is supported efficiently on modern hardware, and the overall per input sample update time remains of the same order as for the basic resonator in reference implementations, several orders of magnitude better than real-time.

Figure 1 shows plots of the magnitude, phase difference and estimated instantaneous frequencies over time in response to a sinusoidal signal of fixed amplitude 1, whose frequency varies around the resonator's resonant frequency, with a 0.05s transition duration. In each case, the instantaneous frequencies computed by the resonator correctly track the actual input signal's frequency. As should be expected, convergence takes longer for lower frequencies, as the resonator dynamics tuning must (and do) take into consideration the laws of physics. The phase time derivative estimate is proportional to the difference between the resonator's resonant frequency and the input sinusoid's frequency (the property used for instantaneous frequency estimation), but so is the magnitude of the resonator. Indeed, for a fixed input signal amplitude, the resonator's response is (by design) maximal if the input's frequency is the same as the resonant frequency, and decreases with the distance between input and resonant frequencies. This is evidenced for example by the peaks in the magnitude plots that coincide with the second input frequency transition, as the input frequency value first moves towards the resonant frequency and then away. Consequently, correctly estimating the actual amplitude of the input signal component is not as straightforward as computing the magnitude of the resonator, but rather should in addition take into account the instantaneous frequency estimate [5].

### 3.2 Resonator Banks

Banks of resonators as described above, independently tuned to perceptually relevant frequency scales, compute an instantaneous estimate of the spectral content of the input signal, including not only magnitudes but also phase time derivative and instantaneous frequency estimates. The overall update computation time for each input sample remains of the same order as for the basic resonator bank (without instantaneous frequency computation overhead) in reference implementations, well within real-time performance on modern hardware.

The plots in Figure 2 show steady state magnitudes for banks of resonators tuned at geometrically spaced frequencies, in response to a sinusoidal signal of fixed amplitude (1) and frequency (1000 Hz). The bottom graph plots the opposite of each resonator's magnitude at the resonator's resonant frequency position on the x-axis; the top graph plots the resonators' magnitudes at the instantaneous frequency position. Resonators that have significant magnitude, i.e. those whose resonant frequency is in the vicinity of the input signal's frequency, agree on the correctly estimated instantaneous frequency. For the purpose of estimating instantaneous frequencies, the actual resonant frequencies in the bank become irrelevant, as long as their density and distribution resonably covers the range of interest. For musical application, a density of 12 resonators per octave (corresponding to 1 semi-tone intervals) covering the range 20-20000 Hz seems reasonable. As illustrated in Figure 2b, a much sparser bank performs just as well in the case of a single frequency input.

The relevant resonators have different magnitudes however, as the distance from the input frequency and each resonator's resonant frequency is different (by construction). The resonator whose resonant frequency is closest to the actual input frequency has the largest magnitude (clos-

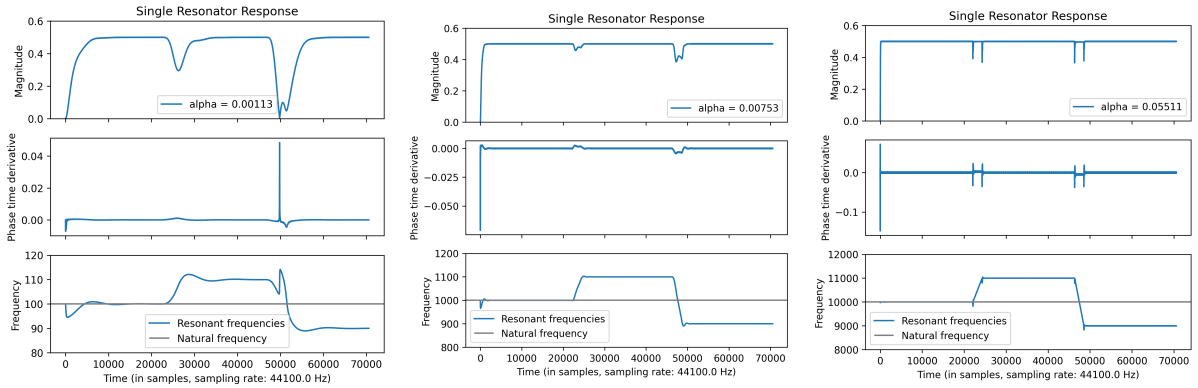


**Figure 3**. Magnitude, phase difference and resonant frequency over time in response to a sinusoidal signal of fixed amplitude 1 whose frequency varies around the tracking resonator's natural frequency (transition duration 0.05s).

est to the correct value for the unit amplitude input), only directly proportional to the input signal's component amplitude if the resonator's resonant frequency matches the input's instantaneous frequency. The next section builds on this observation to expand the model in a direction resolutely inconceivable with FFT-based approaches.

## 4. FREQUENCY TRACKING

In FFT-based phase vocoders, the analysis buffer length and the input signal sampling rate dictate the density and distribution of frequency bins. By construction, frequency bands remain constant. The *Resonate* model imposes no such constraint; its dynamic nature and efficiency make the frequency tracking approach developed in this section possible and practical.

### 4.1 Frequency Tracking Resonator

The frequency tracking resonator expands the basic resonator model by allowing the resonant frequency to change over time: in the absence of significant information (i.e. below a set magnitude threshold), the resonant frequency remains constant, equal to the resonator's natural frequency. In the presence of significant response, however, the resonant frequency tracks the estimated instantaneous frequency.

Resonant frequency tracking is easily achieved by substituting the adaptive phasor of Equation 9 for the fixed angular velocity phasor of Equation 1. Angular velocity changes are stabilized by applying an EWMA with time constant $\gamma_k$. In practice, $\gamma_k = \alpha_k/2$ yields satisfactory results. The remainder of the model update logic remains unchanged. From a computational complexity point of view, this adaptation does not add expensive operations, and performance remains well within real-time limits in reference implementations on modern hardware.

$$\omega_k(t) = \omega_k(t - \Delta t) + \gamma_k \frac{\Delta\phi_k(t)}{\Delta T}$$
$$P_k(t) = P_k(t - \Delta t)e^{-i\omega_k(t)\Delta t} \quad (9)$$

Strictly speaking, a change in the resonant frequency should also trigger a consistent update in the dynamics parameters $\alpha_k$, $\beta_k$ and $\gamma_k$. However if the tracked frequency remains in the vicinity of the resonator's frequency, such an update is not necessary in practice.

Figure 3 shows plots of the magnitude, phase difference and resonant frequencies over time in response to the same sinusoidal signals as those of Figure 1. As the input frequency changes, the system's resonant frequency tracks the estimated input frequency. Consequently, the phase time derivative remains close to 0, and the resonator's magnitude remains close to the maximum value (0.5 for an input signal of amplitude 1).

Here again, convergence takes longer for lower frequencies. However, convergence time from one frequency to another should be inversely proportional to the distance between the two frequencies. A resonator whose natural frequency is closer to the input frequency should converge faster in response to a step signal. Tracking of a smoothly changing frequency depends on the rate of change in the input frequency and the resonator's dynamics, but should be almost instantaneous.

### 4.2 Self-tuning Resonator Bank

A bank composed of frequency tracking resonators constantly self-tunes to the contents of the input signal.

The overall update computation time for each input sample remains of the same order as for the basic resonator bank in reference implementations, well within real-time performance on modern hardware. As an informal reference point, a C++ implementation that makes use of concurrency consistently updates a bank of 112 resonators in well under 2 $\mu$s on platforms ranging from iPad A16 to M4Pro MacBook Pro. Vectorized implementations taking advantage of Single Instruction Multiple Data (SIMD) architectures where available bring the per-sample update duration down to under 650 ns on the same platforms. The open source package `Oscillators`[1] offers Swift and C++ reference implementations of the algorithms described in this paper.

The plots in Figure 4 show steady state magnitudes for banks of resonators tuned at geometrically spaced frequencies, in response to a sinusoidal signal of fixed amplitude (1) and frequency (1000 Hz). The bottom graph plots the opposite of each resonator's magnitude at the resonator's natural frequency position on the x-axis; the top graph plots the resonators' magnitudes at the resonator's (tracked) resonant frequency.

[1] `github.com/alexandrefrancois/Oscillators`

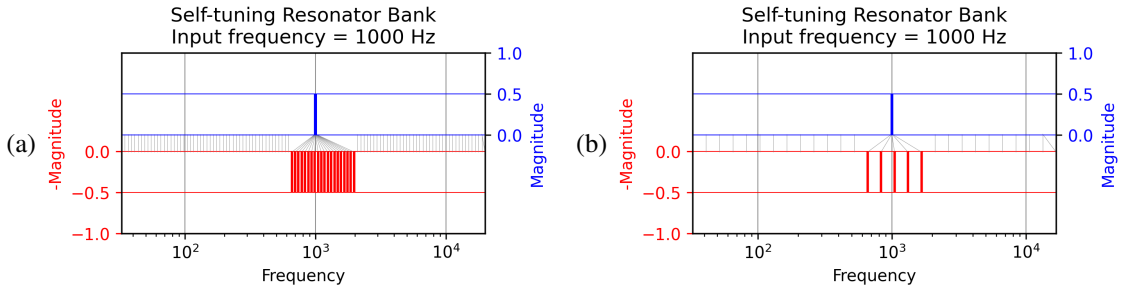


**Figure 4**. Steady state magnitudes for banks of tracking resonators tuned at geometrically spaced natural frequencies, in response to a sinusoidal signal of fixed amplitude (1) and frequency (1000 Hz). The bottom graph plots the opposite of each resonator's magnitude at the resonator's natural frequency position on the x-axis; the top graph plots the resonators' magnitudes at the resonator's (tracked) resonant frequency. (a) 112 frequencies from 32.7 to 19910.2 Hz, (b) 28 frequencies from 32.7 to 16742.4 Hz. Sampling rate: 44100 Hz.

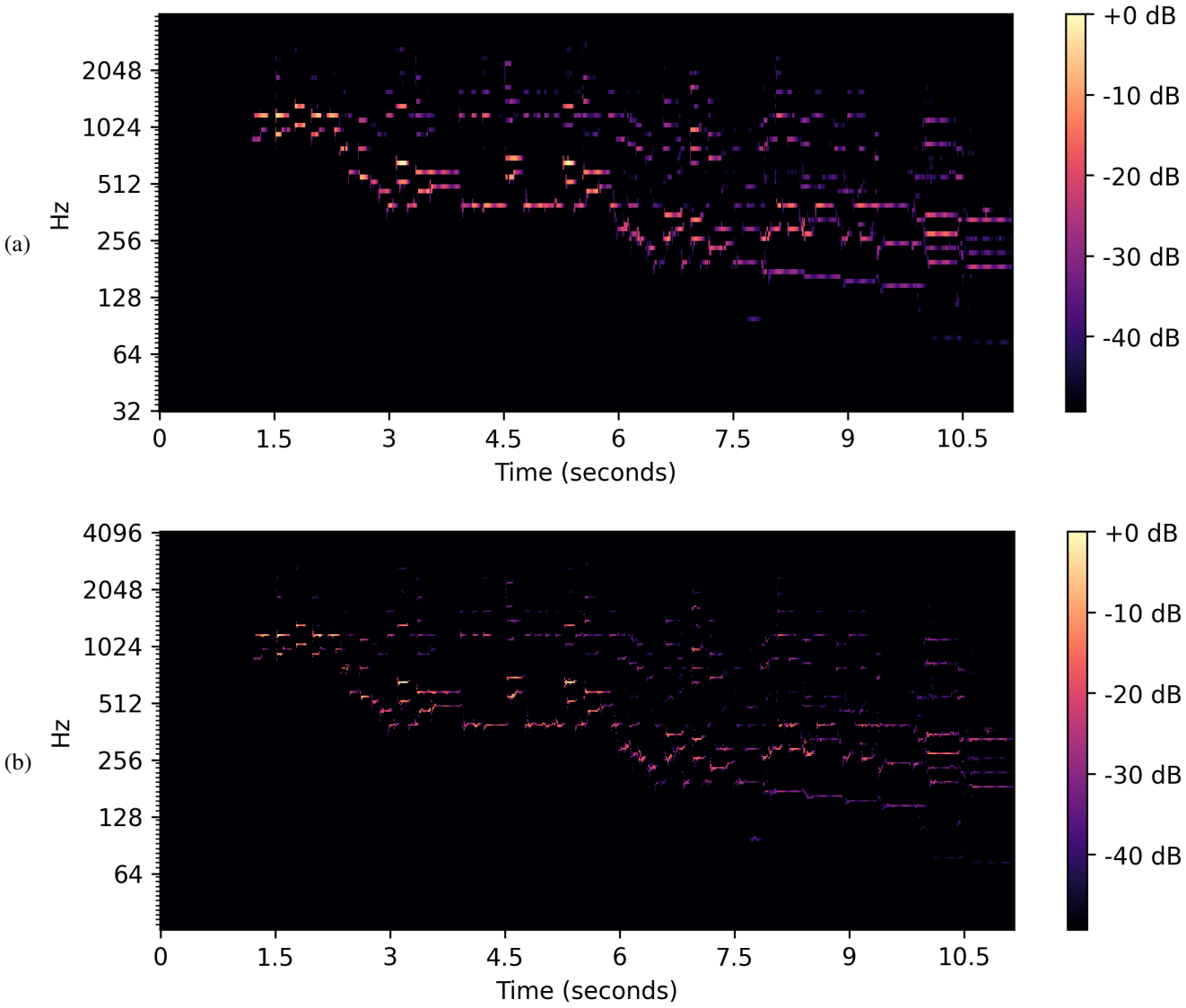


**Figure 5**. Tracking power spectrogram of a short musical excerpt (first 11.15s of *Who's Loving You* by The Jackson 5; electric piano). Sampling rate: 44100 Hz, 84 resonators with geometrically spaced natural frequencies in the range 32.7-3950.7 Hz, display hop length: 512 samples, 84 (a) and (b) 252 (b) display lines.

As the system adapts, the magnitude of each resonator remains close to the maximum. When processing a more natural signal, the magnitude of the resonators will track the amplitude of the frequency component in the input signal. With this approach, the actual tuning (natural frequencies) of the resonators in the bank does not matter, as long as the bank offers an appropriate coverage of the frequency range of interest. The bank continuously self-tunes to the content of the input signal. As already observed, optimal density and distribution of natural frequencies may vary for different applications. Figure 4b illustrates how a much sparser bank can satisfactorily adapt to at least a single frequency input signal.

As nearby resonators converge towards the same frequency tracking the input signal's dominant component frequency in a given region, the output of the resonator whose natural frequency (from which the dynamics parameters are set) is closest to the tracked frequency is given precedence. Consequently, the dynamics of the resonators need not be updated as a function of the tracked frequency. The outputs from the other overlapping resonators are discarded, as they all converge to the same frequency and magnitude and are therefore redundant. Let $\mathcal{T}(t)$ denote the set of indices of the tracking resonators thus selected at each time.

Analysis of an audio signal by a self-tuning resonator bank as described here produces, in real-time, for each input sample, a list of uniquely identified and precisely tracked frequency components present in the input signal, together with their correct amplitude. The result is a spectral model of the input signal exhibiting high temporal resolution (input signal sample rate), high frequency resolution and high amplitude accuracy.

Note that the computations also include the phase of the tracking components; correct preservation of temporal phase coherence across component handover is not critical for analysis purposes, but a crucial area of future improvements for synthesis (see Section 5).

### 4.3 Super-resolution Tracking Spectrograms

To illustrate the output of the self-tuning resonator bank processing of real signals in a familiar format, Figure 5(a) shows a power spectrogram computed off-line in a Python environment and rendered with Librosa's [10] `specshow` function. For each rendered time slice, the output of each tracking resonator is rendered in the line corresponding to its current resonant frequency. The lines correspond to the logarithmically distributed natural frequencies of the bank's resonators (12 lines per octave). In case of tracked frequency overlap between nearby resonators, the value at the line is set to the maximum power. (This is equivalent to keeping the power of the resonator whose resonant frequency is closest to its natural frequency.) Such a spectrogram is however a low fidelity rendering of the available instantaneous frequency information, which is computed with much higher accuracy. Figure 5(b) shows the same data rendered as a spectrogram with triple the original frequency resolution (36 lines per octave). In both cases, the display hop length of 512 samples does not reflect the high temporal resolution of the computed data, which comprises one estimate per input sample.

Figures 6 and 7 show screenshots of power spectrograms of musical passages, computed and rendered in real-time using the Oscillators app [2], with a display hop length of 64 samples, to better illustrate the high temporal resolution of the computed spectral representation. The number of frequency lines in the spectrograms is triple the number of resonators in the bank. Still images of such spectrograms necessarily cover shorter durations of content (the Resonate website links to video captures of real-time demonstrations) but reveal more details. Figure 6 features solo violin with fast notes, vibratos and glissandi (from *Paganini's Caprices*). Figure 7 illustrates fine harmonic content tracking and capture of rhythmic transients in rock music (Queen's *Under Pressure* and *Another One Bites The Dust*).

The information computed seems suitable for real-time, low-latency MIDI event generation from the input signal, a promising potential application currently under investigation. Yet, these visual representations also highlight the limitations of a fixed-size grid to represent and use the output of the proposed model, which consists, for each input sample, of a variable size list of arbitrary frequency-amplitude pairs. In particular, optimal use of such dynamic unstructured data may call for a different architecture than for example the prevailing fixed-size tensor that underlies current machine learning architectures.

## 5. SYNTHESIS

The traditional phase vocoder relies on the iFFT for the synthesis phase, so that in the absence of parameter modifications, the re-synthesized signal is identical to the input signal. Possible modifications include frequency shift and time modification. A similar transform can be achieved with *Resonate*, albeit without the identity property, which would require impractical additional constraints on the frequency density and distribution in the resonator bank. Applying the inverse phasors of Equation 10 to individual tracking resonators $\tilde{R}_k(t)$ yields the complex numbers $S_k(t)$ of Equation 11, from which an audio signal corresponding to the input signal's component can readily be produced. The reverse phasor can optionally introduce a frequency shift ratio $fs$ and a different target sampling rate $sr_s = 1/\Delta t_s$ (time manipulation). A frequency ratio of 1 keeps the frequency unchanged, values greater or lower than 1 result in higher or lower frequency, respectively. A sampling rate equal to the input's sampling rate keeps the timing/speed, a greater or lower sampling rate results in a shorter or longer duration output signal, respectively.

$$P_k^{-1}(t) = P_k^{-1}(t - \Delta t_s)e^{i\omega_k(t) fs \Delta t_s} \tag{10}$$

$$S_k(t) = \tilde{R}_k(t) P_k^{-1}(t) \tag{11}$$

Equation 12 produces an audio signal $s(t)$ which captures salient features of the input signal. The signal is constructed from non overlapping tracking resonators, so that individual components occur only once in the sum. Consequently, vertical phase coherence is not a concern and there is no need for phase locking [6]. However, horizontal phase discontinuities will occur when the dominant resonator in a frequency region changes, potentially causing some clicks in the output signal.

[2] `alexandrefrancois.org/Oscillators`

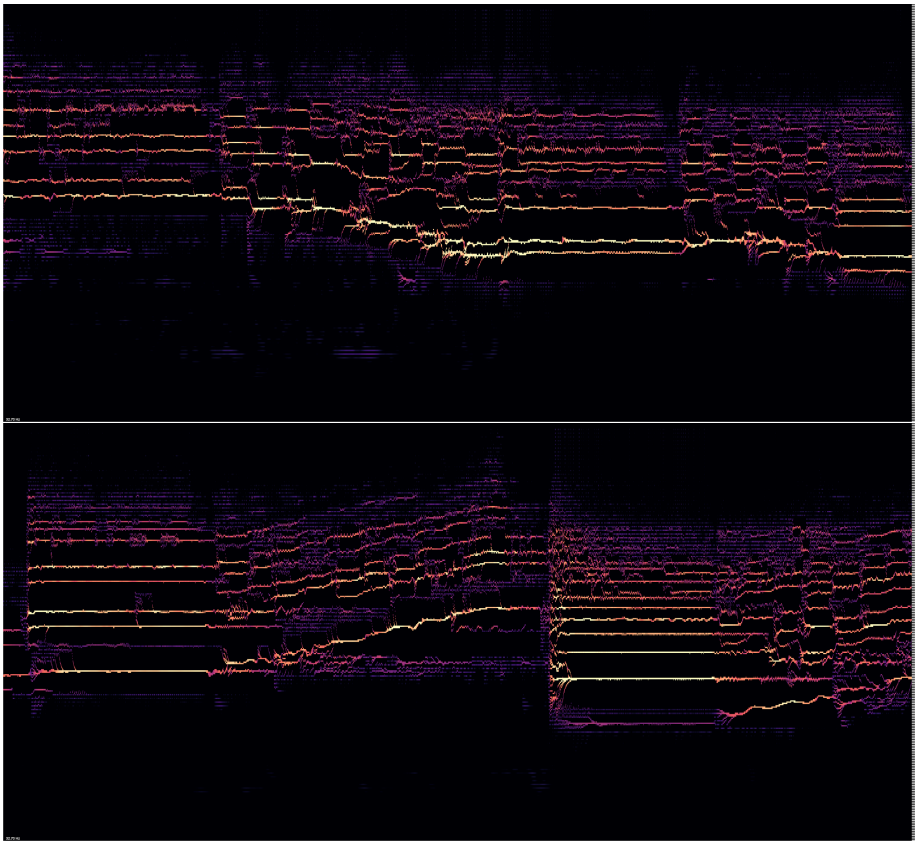

**Figure 6**. Tracking power spectrogram of short solo violin excerpts from the *Paganini Caprice no.24 in A Minor*. Sampling rate: 44100 Hz, 112 resonators, with geometrically spaced natural frequencies in the range 32.7 to 19910.2 Hz, display hop length: 64 samples, 336 display lines.

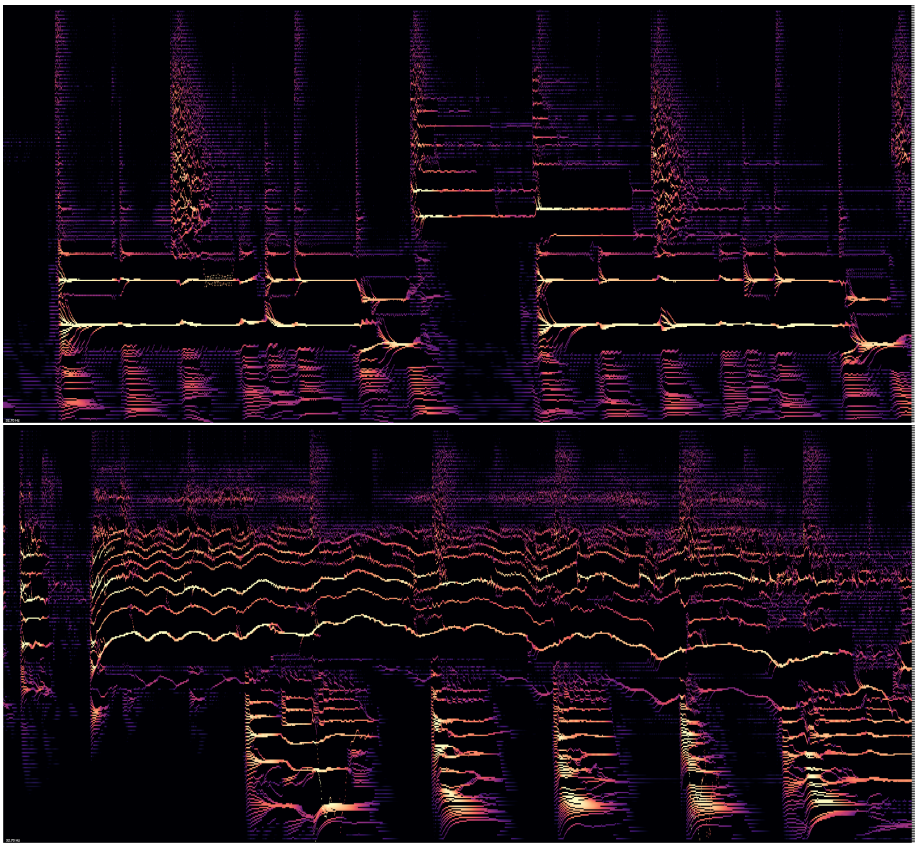

**Figure 7**. Tracking power spectrogram of short excerpts from Queen's *Under Pressure* (top) and *Another One Bites The Dust* (bottom). Sampling rate: 44100 Hz, 112 resonators, with geometrically spaced natural frequencies in the range 32.7 to 19910.2 Hz, display hop length: 64 samples, 336 display lines.

$$s(t) = KRe(\sum_{k \in \mathcal{T}(t)} S_k(t)) \quad (12)$$

A complete study and evaluation of synthesized signals is out of the scope of this paper, but early experiments with and without frequency shifts and time modifications are encouraging. See the Resonate project website [3] for examples.

Informally, the synthesized audio seems to capture the harmonic content of the input signal. It also exhibits transient smearing (due to the dynamics of the frequency tracking) and some other artifacts likely due to the partial nature of the reconstruction. In these first experiments, the clicks caused by horizontal phase discontinuities are mediated with simple smoothing. If more research is needed to improve the quality of the synthesized audio, the method seems promising with possible applications in real-time accompaniment and improvisation systems, for example.

## 6. SUMMARY AND FUTURE WORK

This paper described an original approach to computing a high temporal and frequency resolution, and high amplitude accuracy spectral representation of audio signals, in real-time and with low latency.

Applying techniques from phase vocoders to make use of phase information, a new tracking resonator model extends the original *Resonate* model while retaining its efficiency and characteristic properties. A bank composed of frequency tracking resonators constantly self-tunes to the contents of the input signal. With this approach, the actual tuning of the resonators in the bank does not matter, as long as the bank offers an appropriate coverage of the frequency range of interest for the target application.

The extended model formulation builds on the same compact, iterative approach, which affords computing an update at each signal input sample, requiring no buffering and involving only a handful of arithmetic operations, and adding one single transcendental function computation per resonator. Straightforward reference implementations exhibit real-time performance on modern hardware, while vectorized parallel implementations taking advantage of SIMD architectures further reduce computation time, and therefore latency.

Self-tuning banks form the basis for an analysis technique that produces, in real-time, for each input sample, a list of uniquely identified and precisely tracked frequency components present in the input signal, together with their correct amplitude. High temporal and frequency resolution spectrograms illustrated the analysis result of real musical signals in a familiar format.

The detailed representations produced can potentially improve the quality and accuracy of any traditional applications, for example in Music Information Retrieval, tonal analysis and visualization to name a few. They also offer promising prospects for real-time, low-latency applications such as accompaniment and improvisation systems, via real-time audio synthesis or MIDI event generation. Encouraging initial synthesis experiments, with and without time and frequency manipulations, invite further investigation and explorations.

The results presented here remain preliminary and more work is needed to improve the quality of the analysis and synthesis outputs. Optimal use of the dynamic unstructured data produced may require a departure from, for example, the prevailing fixed-size tensor that underlies current machine learning architectures. Another interesting potential area for future research is the application of machine learning techniques to optimize the values of model parameters, such as the density and distribution of natural frequencies in self-tuning banks. To facilitate wider community contributions to the development of the model, in addition to resources already referenced above, the open source module `noFFT` [4] provides python and C++ implementations of *Resonate* functions and Jupyter notebooks with the code used to produce the figures in the paper.

## 7. REFERENCES


[1] A. R. François, "Resonate: Efficient Low Latency Spectral Analysis of Audio Signals," in *Proceedings of the International Conference on Computer Music*, Boston, MA, USA, 2025, pp. 251–258.

[2] M. R. Portnoff, "Implementation of the digital phase vocoder using the fast Fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-24, no. 3, pp. 243–248, 1976.

[3] J. A. Moorer, "The Use of the Phase Vocoder in Computer Music Applications," *Journal of the Audio Engineering Society*, vol. 26, no. 9, pp. 717–727, 1978.

[4] M. Dolson, "The Phase Vocoder: A Tutorial," *Computer Music Journal*, vol. 10, no. 4, pp. 14–27, 1986.

[5] D. W. Griffin and J. S. Lim, "Signal estimation from modified short-time Fourier transform," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 1983, pp. 804–807.

[6] M. Puckette, "Phase locked vocoder," in *Proceedings of the 1995 International Computer Music Conference (ICMC)*, 1995, pp. 245–248.

[7] J. C. Brown and M. Puckette, "A High-Resolution Fundamental Frequency Determination Based on Phase Changes of the Fourier Transform," *Journal of the Acoustical Society of America*, vol. 94, no. 2, pp. 662–667, 1993.

[8] J. Laroche and M. Dolson, "Improved phase vocoder time-scale modification of audio," *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 3, pp. 323–332, 1999.

[9] Z. Průša and N. Holighaus, "Phase Vocoder Done Right," in *25th European Signal Processing Conference (EUSIPCO)*. IEEE, 2017, pp. 1–5.

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


[3] `alexandrefrancois.org/Resonate`

[4] `github.com/alexandrefrancois/noFFT`