Birds on Holiday

Nathan Turner CMath FIMA, QinetiQ

Domestic holidays in the UK offer a great opportunity to escape the everyday without the challenges of packing lists, airports and refilling miniature shampoo bottles! Furthermore, we have the chance to explore the extensive variety of British landscapes and their ecosystems.

While exploring the outdoors, particularly in wooded or wetland areas, we can also sample the beautiful array of sounds offered by local birds. Whatever your experience with identifying birds, a great resource available for free on your smartphone is Merlin Bird ID (`merlin.allaboutbirds.org`), which allows birds to be identified by their song.

How would you go about taking an audio recording and correctly identifying the associated bird? We will explore the mathematics underpinning the signal processing required to do this and consider the effectiveness of two methods for preparing the data for classification algorithms.

Throughout this article, you can listen along to related audio clips by visiting the Macaulay Library website (`macaulaylibrary.org`). You may find it interesting to listen to the audio while inspecting the spectrogram visualisations. In addition, you can explore the parameters of the analysis yourself by downloading the code and audio from the GitHub repository (`github.com/nturnermaths/Birding`), where you can also find high-resolution graphical outputs.

The short-time Fourier transform

Given a time series signal y, the continuous infinite Fourier transform is defined as

$$\mathcal{F}(y) = \int_{-\infty}^{\infty} y(t)\, \mathrm{e}^{-\mathrm{i}(2\pi f)\tau}\, \mathrm{d}\tau. \tag{1}$$

However, for real audio signal processing, we need to translate finite-duration time-domain signals into the frequency domain so that we can understand their spectral content, which may also vary with time.

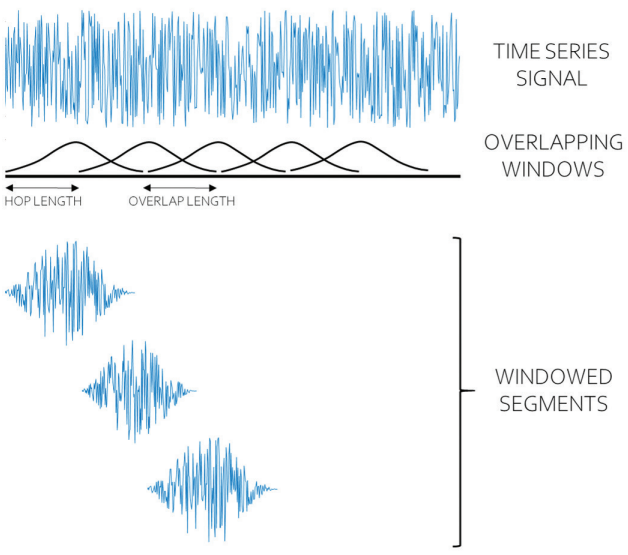


Figure 1: Illustration of STFT windowing

© Bob Brewer | Unsplash

Figure 2: European goldfinch perched on a branch

To do this, we use the discrete Fourier transform (DFT):

$$Y(k) = \sum_{n=0}^{N-1} y(n)\, \mathrm{e}^{-\mathrm{i}(2\pi k)n/N}, \tag{2}$$

where $k \in [0, N-1]$ is the discrete frequency, N is the number of samples and $y(n)$ is the signal value for sample n. The output is then a discrete array of complex values across the frequency domain, which practically approximates the Fourier transform for a finite time series signal.

If we wish to understand how the spectral (frequency) content of a signal changes across the duration of the signal, we compute multiple DFTs. This approach is known as a short-time Fourier transform (STFT):

$$Y(k,t) = \sum_{n=0}^{N-1} y(n+tH)w(n)\, \mathrm{e}^{-\mathrm{i}(2\pi k)n/N}, \tag{3}$$

where t represents the time step index of the STFT, H is the hop size and w is a window function of length N. The output is now a discrete array of complex values across the time and frequency domains.

The window function can take several forms and is applied to the signal to avoid spectral leakage from discontinuities in the segmented time series data. The hop size is the number of samples the window shifts over for each frame of the STFT; this is usually set such that the frames overlap, which ensures that no time-domain information is lost. This procedure is illustrated in Figure 1.

A typical choice for windowing parameters in audio signal processing is 75% overlap and, as adopted in this analysis, a periodic (sinusoidal) Hann window of length 128 samples.

To visualise the STFT of a time signal, we plot the magnitude $Y(k,t)$ against both time and frequency. These plots are referred to as spectrograms, and they show how the amplitude (colour intensity) of the output changes with both time (x-axis) and frequency (y-axis).

European goldfinch example

The European goldfinch (Figure 2) is a beautiful bird, visually identified by its striking yellow wing patch, black tail and cherry red face. You might see them acrobatically foraging seeding plants, skillfully putting their pointed bills to good use. We will use a goldfinch song recording[1] as an example. This goldfinch song contains many trills, fast clicks and sweeping sounds, making it perfect for this visualisation exercise.

We adopt the STFT method to analyse an 11-second audio recording while varying the processing parameters to assess their impact on the method. The frequency resolution f_{STFT} of the STFT is related to the choice of DFT length L and the sample frequency resolution f_s according to

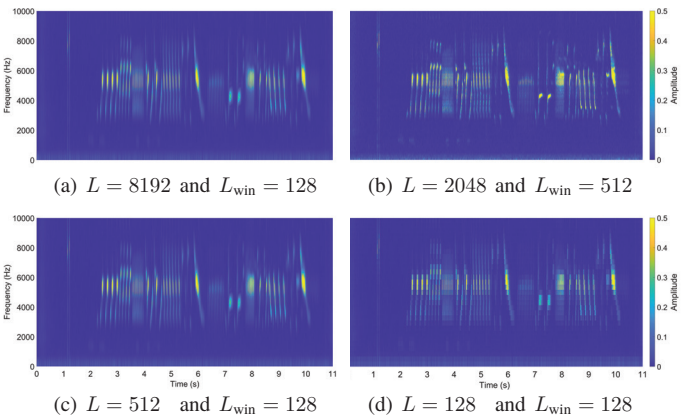
$$f_{\text{STFT}} = f_s/L. \tag{4}$$

With a sample frequency f_s of 44 100 Hz (a typical audio sample frequency for smartphones), equation (4) then gives the following STFT resolutions for the spectrograms shown in Figure 3, which compares four combinations of DFT length L and window length L_{win}:

(a) $L = 2^{13} = 8192 \implies f_{\text{STFT}} \approx 5\,\text{Hz}$ with $L_{\text{win}} = 128$,
(b) $L = 2^{11} = 2048 \implies f_{\text{STFT}} \approx 22\,\text{Hz}$ with $L_{\text{win}} = 512$,
(c) $L = 2^{9} = 512 \implies f_{\text{STFT}} \approx 86\,\text{Hz}$ with $L_{\text{win}} = 128$,
(d) $L = 2^{7} = 128 \implies f_{\text{STFT}} \approx 345\,\text{Hz}$ with $L_{\text{win}} = 128$.

Increasing the DFT length improves the frequency resolution of the STFT output, leading to greater fidelity in the (frequency) y-axis. Compare, for example, spectrograms (a) and (d). However, increasing the DFT length also necessitates greater processing capability (to perform the analysis in a fixed time).

Therefore, if we wish to apply a real-time classification algorithm, we need to balance the fidelity of the output against the processing power and analysis time requirements. There are, however, many techniques to improve the performance of classification algorithms that are not memory intensive.



(a) $L = 8192$ and $L_{\text{win}} = 128$ (b) $L = 2048$ and $L_{\text{win}} = 512$

(c) $L = 512$ and $L_{\text{win}} = 128$ (d) $L = 128$ and $L_{\text{win}} = 128$

Figure 3: Spectrograms of goldfinch recording

First, we will explore adjusting the STFT window length. For a fixed window overlap, increasing the STFT window length reduces the number of time steps in the STFT, which reduces the fidelity of the (time) x-axis. This also reduces the processing time, so that we can tune the DFT length and STFT window length together to balance the processing capability with both the frequency and time resolution.

For example, the frequency resolution of spectrogram (b) is a factor of 4 better than that for spectrogram (c), but (b) was produced using an extended window length. This sacrifices some time resolution but maintains the same processing time without losing the main temporal characteristics of the signal.

When classifying birds from recorded audio, it is typical to tune the parameters such that the time resolution is less than or equal to the minimum syllable length of an individual chirp (about 50 milliseconds). In our example, spectrogram (b) achieves a time resolution of 3 milliseconds, while (c) has a time resolution of $<$1 millisecond. This suggests that the settings used for (b) would be more than sufficient to balance the temporal and spectral characteristics of interest.

Continuous wavelet transforms

To overcome the challenges of balancing the time and frequency resolution, we can consider alternative transforms. The continuous wavelet transform is one such alternative for a simultaneous time and frequency analysis.

As detailed in [1], the continuous wavelet transform (CWT) of a time series signal x is defined as

$$W_\psi(t,s) = \int_{-\infty}^{\infty} \frac{1}{s}\psi^*\left(\frac{\tau - t}{s}\right) x(\tau)\,\mathrm{d}\tau, \tag{5}$$

where $\psi(t)$ is the wavelet, $*$ represents the complex conjugate, t is time and s is a scale parameter (analogous to frequency). Each wavelet transform is defined for multiple scales s. Equation (5) shifts each scaled wavelet along the time series, resulting in coefficients for each wavelet scale at every time sample from the original signal (analogous to the sinusoidal decomposition in the Fourier transform). This avoids the time sampling challenges faced by STFT processing.

A wavelet function ψ must have zero mean, finite energy and satisfy the admissibility condition:

$$\int_{-\infty}^{\infty} \frac{|\Psi(\omega)|^2}{|\omega|}\,\mathrm{d}\omega < \infty, \tag{6}$$

where $\Psi(\omega)$ is the Fourier transform of ψ and $\omega = 2\pi f$ is the angular frequency. Wavelets that vanish for negative frequencies (i.e., $\Psi(\omega) = 0$ for $\omega < 0$) are analytic. For such transforms, (5) defines the analytic wavelet transform, which can be represented in the frequency domain as

$$W_\psi(t,s) = \frac{1}{2\pi}\int_0^{\infty} \Psi^*(s\omega)X(\omega)\,\mathrm{e}^{\mathrm{i}\omega t}\,\mathrm{d}\omega, \tag{7}$$

where X is the Fourier transform of the signal x.

Wavelets are commonly defined as parameterised families. In this example, we will consider the Morse wavelet, defined in the frequency domain by

$$\psi_{\beta,\gamma}(t) \iff \Psi_{\beta,\gamma}(\omega) = H(\omega)a_{\beta,\gamma}\omega^{\beta}\,\mathrm{e}^{-\omega^{\gamma}}, \tag{8}$$

where H is the Heaviside function and $a_{\beta,\gamma}$ is a normalising constant. The time–bandwidth product $\beta > 0$ and symmetry $\gamma > 0$ parameterise the shape of the wavelet.

By its very definition, $\Psi_{\beta,\gamma}$ has a peak angular frequency. We can determine this by finding ω_{Pk} such that

$$\frac{\mathrm{d}\Psi}{\mathrm{d}\omega}(\omega_{Pk}) = 0. \tag{9}$$

The natural logarithm of $\Psi_{\beta,\gamma}$ is more readily differentiated. For $\omega > 0$:

$$\ln(\Psi_{\beta,\gamma}) = \ln(a_{\beta,\gamma}) + \beta \ln(\omega) - \omega^{\gamma}, \tag{10}$$

$$\frac{\mathrm{d}}{\mathrm{d}\omega}(\ln(\Psi_{\beta,\gamma})) = \frac{\beta}{\omega} - \gamma\omega^{\gamma-1}. \tag{11}$$

This gives $(\omega_{Pk})^{\gamma} = \beta/\gamma$, so the peak angular frequency is

$$\omega_{Pk} = \left(\frac{\beta}{\gamma}\right)^{1/\gamma}. \tag{12}$$

By way of example, setting $\beta = 60$ and $\gamma = 3$ gives a peak angular frequency $\omega_{Pk} \approx 2.71$, or equivalently for our audio signal,

$$f_{Pk} = \frac{\omega_{Pk} f_s}{2\pi} \approx 19\,052. \tag{13}$$

When undertaking a CWT analysis, the frequency domain is subdivided exponentially, each frequency being determined by the scale s. This is parameterised by the number of voices per octave; the number of filters for each doubling of frequency. The scale spacing is, therefore, defined by

$$s(j) = 2^{j/v} s_0, \tag{14}$$

where j is the scale index, $10 \leq v \leq 48$ is the number of voices per octave and s_0 is the base scale. Choosing $v = 48$, the scale spacing is $2^{1/48} \approx 1.0145$. This represents a high value for v and gives a fine frequency-domain spacing.

An example Morse wavelet $\psi_{60,3}$ is shown in Figure 4. This shows the real and imaginary parts, with the magnitude of the wavelet in the time domain for scale $s = 700$ (the total number of scales in this analysis is 770).

The exponential spacing of the frequency domain gives rise to fine resolution at low frequencies and coarse resolution at high frequencies. This, together with the finer resolution temporal output, makes the CWT well suited to fast transient sounds (3.5–4.0 seconds in the goldfinch recording) and 'sweeps', where a continuous sound varies in frequency over time (6.0 and 10.0 seconds in the goldfinch song recording).

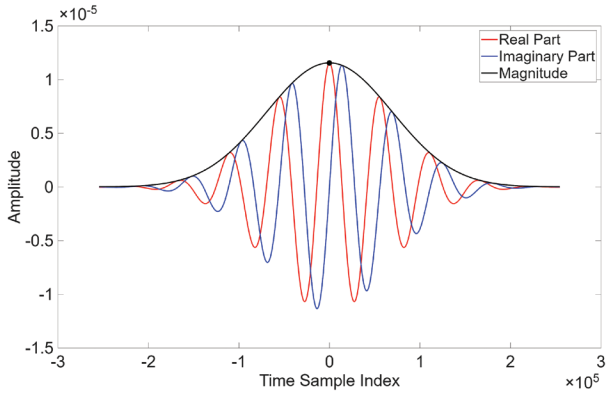


Figure 4: Morse wavelet: frequency scale $s = 700$ with 48 voices per octave.

We can visualise the output of the CWT via a scalogram. This is like the spectrogram we encountered earlier, but it has a logarithmic y-axis to account for the non-uniform frequency scaling. Comparing Figure 5 with spectrogram (a), greater detail is revealed for fast transients and sweeps than realised with the STFT. The colour intensity scales of the spectrogram and scalogram have been set to ensure saturation at the higher amplitudes, which improves visualisation of the signals.

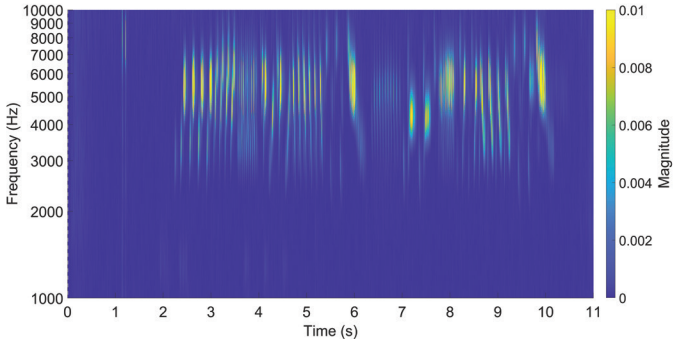


Figure 5: Scalogram of goldfinch recording (Morse [60,3], 48 voices per octave).

Conclusion

We have explored the mathematical foundations of two distinct signal processing methods, both of which enable visualisation of recorded audio signals. Taking this further, one could explore the benefits of each method for bird classification from recorded audio.

Convolutional neural networks are commonly used to identify patterns in two-dimensional visualisations, such as spectrograms and scalograms. The CWT has been shown to be more robust than the STFT when used as input to some classification tasks, for example, recognising faults in machinery from non-stationary acoustic signals [2].

Application of the CWT typically necessitates increased computational time (or additional processing capability) relative to the STFT. It is important to consider the processing time requirements; for example, an STFT-based method may be more appropriate for a remotely deployed near-real-time classification algorithm, whereas a CWT-based approach may yield more reliable results if an a posteriori analysis is acceptable.

Notes

1. European goldfinch audio used with permission from Gregory Budney and the Macaulay Library, see `macaulaylibrary.org/asset/181668`.

References

1 Lilly, J.M. and Olhede, S.C. (2010) On the analytic wavelet transform, *IEEE Trans. Inf. Theory*, vol. 56, no. 8, pp. 4135–4156.

2 Phan, D.T. (2025) Comparison of short-time Fourier transform and wavelet transform as input data to convolutional neural networks for acoustical machinery fault recognition, Master's thesis, Berliner Hochschule für Technik.