Idealised vs. Practical Filters

- Filters are often described in idealised terms, e.g. as ‘brickwall’ filters
- In reality, practical filters approximate the idealised forms
Practical Filters

**Lowpass**

- **Gain**
- **Frequency**
- **Passband**
- **Ripple**
- **Stopband**
- **Attenuation**

**Bandpass**

- **Gain**
- **Frequency**
- **Passband**
- **Stopband**
- **Ripple**
- **Slope**
- **Stopband Attenuation**
Filter Design

- The behaviour of a filter is defined by the number and the location of its 'poles' and 'zeros' (as we saw in Lecture #16).
- The complexity of a filter is often characterised by its 'order'.
- In a digital filter, order is defined as the number of past (input or output) values that are involved in the calculation.
- For example, a filter that takes two past values is termed a 'second-order filter'.
- Filter order can thus be related to the number of poles and zeros.
- Hence, the degree to which a practical filter approximates the idealised form is a function of its order (i.e. the number of poles and zeros).
General Difference Equation

- The general difference equation may include any number of past inputs and outputs …

\[ y[nT] = a_1 y[(n-1)T] + a_2 y[(n-2)T] + \ldots + a_p y[(n-p)T] + b_0 x[nT] + b_1 x[(n-1)T] + b_2 x[(n-2)T] + \ldots + b_q x[(n-q)T] \]

- Taking Z transforms …

\[
Y(Z) = Y(z) \sum_{i=0}^{p} a_i z^{-i} + X(z) \sum_{i=0}^{q} b_i z^{-i}
\]

\[
H(z) = \frac{\sum_{i=0}^{q} b_i z^{-i}}{1 - \sum_{i=0}^{p} a_i z^{-i}}
\]
General Digital Filter

- A very wide range of transfer functions (and associated frequency responses) may be obtained by appropriate choice of the filter parameters
- \( p \) = number of poles, \( q \) = number of zeros
- For the case \( p = 0 \), the resulting non-recursive filter is said to be a 'finite impulse response - FIR' filter
- If \( p > 0 \), the resulting recursive filter is said to be an 'infinite impulse response - IIR' filter

‘FIR’ Filters

- Finite impulse response (FIR) filters express each output sample as a weighted sum of the last \( N \) inputs (where \( N \) is the order of the filter)
- Advantages
  - inherently stable since they don’t use feedback (i.e. only zeros)
  - the coefficients are usually symmetrical, hence the phase response is linear and signals of all frequencies are delayed equally
  - overflow is straightforward to avoid
  - generally easier to design than IIR filters
- Disadvantages
  - may require significantly more processing and memory resources than the equivalent IIR filter
  - often require a much higher filter order than IIR filters
  - delay can be much greater than for an equivalent IIR filter
‘IIR’ Filters

- Infinite impulse response (IIR) filters are the digital counterpart to analog filters... because they contain an internal state, and the output and the next internal state are determined by the previous inputs and outputs.

- Advantages
  - normally require less computing resources than an FIR filter of similar performance.

- Disadvantages
  - due to the feedback, high-order IIR filters may have problems with instability, arithmetic overflow and limit cycles.
  - careful design is required to avoid such pitfalls.
  - the time delay through such a filter is frequency-dependent (since the phase shift is inherently a non-linear function of frequency).

‘IIR’ Filters

- IIR filters all approximate the ideal brickwall filter ...
  - Butterworth
  - Chebyshev (types I and II)
  - elliptic
  - Bessel

- High-order IIR filters can easily become unstable.
- This is much less of a problem with first and second-order filters.
- 2nd-order IIR filters are often called ‘biquads’.
- Higher-order filters are typically implemented as a cascade of ‘biquad sections’.
‘Biquad’ Filters

- A ‘biquadratic’ (biquad) filter is a 2nd-order recursive linear IIR filter, containing two poles and two zeros.
- The name refers to the fact that its Z domain transfer function is the ratio of two quadratic functions:
  \[ H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \]
- This is derived from the following difference equation:
  \[ y[kT] = b_0 x[kT] + b_1 x[(k-1)T] + b_2 x[(k-2)T] - a_1 y[(k-1)T] - a_2 y[(k-2)T] \]
The Biquad Filter in Pd

- Pd provides a raw biquadratic filter object `[biquad~]`
- The five coefficients \( (b_0, b_1, b_2, a_1, a_2) \) can be sent as a single message
- For a bandpass filter, the coefficients can be calculated using the `[bandpass]` object
- Pd provides the following objects for calculating filter coefficients for `[biquad~]` ...
  - `[lowpass]` `[highpass]`
  - `[bandpass]` `[notch]`
  - `[lowshelf]` `[highshelf]` `[hlshelf]`
  - `[equalizer]`

‘Biquad’ Filters
Filters in Speech Processing

- General filtering
  - notch filtering (to remove interference and noise)
  - pre-emphasis (to equalise the frequency response)

- Modelling speech production
  - the source-filter model (Lecture 3)

- Modelling the auditory system
  - filter-bank analysis (Lecture 4)

Tone Removal by Notch Filtering
Pre-Emphasis

- Averaged over time, speech has an overall spectral slope of \(-6\) dB/octave.
- This means that there can be a large amplitude difference between the lowest (~50 Hz) and highest (~8 kHz) frequency components.
- E.g. over 8 octaves there will be a \((6 \times 8) = 48\) dB difference (which is comparable to the dynamic range from the quietest to the loudest speech sounds).
- This can make processing difficult (especially for functions such as spectral peak picking, i.e. formant tracking).
- It is therefore usual to ‘pre-emphasise’ a speech signal by giving it a \(+6\) dB/octave lift.
- In Pd this can be achieved by placing a ‘real zero’ near the origin, e.g. \([\text{rzero} \sim 1]\).
Auditory Filters

- Filterbanks associated with the DFT/FFT have constant bandwidths and are centered at uniformly spaced locations along the frequency axis.
- To better model the frequency response characteristics of the human ear, many researchers use filters inspired by the auditory system.
- These typically have non-uniform bandwidths and non-uniform spacing of center frequencies.
- The most well-known auditory filter is the ‘gammatone’.
- This filter has an impulse response that is the product of a gamma distribution and sinusoidal tone.

‘Gammatone’ Filter

\[ g(t) = a k^{-1} e^{-\pi t} \cos(2\pi t + \phi) \]


The Gammatone in Pure Data

- Daniel Pressnitzer and Dan Gnansia have implemented a gammatone filterbank in Pd: [gammabank_dm~]
- It is part of an ‘Audition Library’ which contains other useful functions, such as …
  - gammatone re-synthesis
  - multiplexers
  - de-multiplexers
  - outer and middle ear filters
- The code is based on:
- The Audition toolkit is not available on the Lewin Lab. Machines, but is downloadable (for Windows) from: http://lumiere.ens.fr/Audition/tools/realtime/
Gammatone Filterbank

This lecture has covered …

• Idealised vs. practical filters
• Filter design
• Second-order filters
• The general difference equation
• FIR and IIR filters
• The ‘biquad’ filter
• Filters in speech processing
• The ‘gammatone’ filter
Any Questions?

Next time …

Linear Prediction