Recursive or IIR Filters

And why they should be avoided
Recursive Filters

- \( x(n) \rightarrow X(z) \)
- \( x(n-k) \rightarrow z^{-k}X(z) \)
- Why \( z^{-1} \)?
- \( z^{-1} \) corresponds to a unit delay; the basic building block.
- Zeros: Thumbtacks to hold surface down
- Poles: Builds mountains
Different Filter Models

All Zero (FIR) \[ X(z) = a_0 + a_1 z^{-1} + \cdots + a_{N-1} z^{N-1} \]

Pole Zero (IIR) \[ Y(z) = \frac{a_0 + a_1 z^{-1} + \cdots + a_{N-1} z^{N-1}}{b_0 + b_1 z^{-1} + \cdots + b_{M-1} z^{M-1}} \]

All Pole (IIR) \[ Z(z) = \frac{A}{b_0 + b_1 z^{-1} + \cdots + b_{M-1} z^{M-1}} \]
IIR Filter Structures

- The Z transform has an infinite number of terms (denominator polynomial), so there must be some feedback in the filter.

```
x(n+1) x(n) x(n-1)
```

Infinite impulse response
Recursive
Pole-Zero

\[ y(n) = \text{DIRECT IMPLEMENTATION} \]
Linear Phase Filter

Zeros on unit circle in conj pairs
Zeros off unit circle in reciprocal conj pairs
\[ P(z) = z^{-M} P(z^{-1}) \]

Reversal of Coefficients

Huge mountain at origin
Implementation of IIR Filters

If you use direct implementation and the number of poles > 5
Bandwidth/Fs < 0.1
16 bit arithmetic

Then you have built an oscillator
Filters always oscillate in the low order bits.

Roots spread out like point charges due to finite precision.

Works fine in Matlab (80 bit precision) but doesn’t work in practice.
\[ P(z) = \prod_{i} (z - z_i) \]

\[ = \sum a_k z^{N-k} \]

\[ = a_0 z^N + a_1 z^{N-1} + \cdots + a_N \]

Replace \( a_i \) with \( a_i + \Delta a_i \)

\[ \hat{P}(z) = P(z) + \Delta a_i z^{N-l} \]

Need roots \( \Rightarrow \prod_{i} (z - z_i) + \Delta a_i z^{N-l} = 0 \)
Pick one root

\[(z - z_r) \prod_{i \neq r}^i (z - z_i) + \Delta a_i z^{N-l} = 0\]

\[Z = z_r - \frac{\Delta a_i z^{N-l}}{\prod_{i \neq r}^i (z - z_i)}\]

Guess \( Z = z_r \), solve for \( z \), keep substituting

\[Z \approx z_r - \frac{\Delta a_i z_r^{N-l}}{\prod_{i \neq r}^i (z_r - z_i)}\]

Sensitivity
**Sensitivity**

\[
\frac{1}{\prod_{i \neq r} (z_r - z_i)}
\]

If distance = 0.1
Then reciprocal = 10000
Increase sampling by 10 (small bandwidth)
Then reciprocal = \(10^8\)
Roots always spread out like point charges
**Solution**

- Don’t use IIR filters. Stick with polyphase FIR.
  - similar computational efficiency, always stable, low sensitivity to coefficient quantization, linear (any) phase
- Use cascade of low (1st or 2nd) order polynomials
  - similar to analog active filter design
- Alternate structures may reduce stability problems
  - Direct Form (very poor performance)
  - First Canonic Form
  - Second Canonic Form
  - Parallel Form
  - Cascade Form

See Oppenheim & Schafer
2nd Order Canonical Form I

\[ H(z) = b_0 \frac{z^2 + b_1 z + b_2}{z^2 + a_1 z + a_2} \]
Canonical Form I

\[ H(z) = b_0 \frac{z^2 + b_1 z + b_2}{z^2 + a_1 z + a_2} \]

\[ \frac{KY(z)}{X(z)} = \frac{1 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \]

\[ \frac{KY(z)}{X(z)} = \frac{W(z) \ KY(z)}{X(z) \ W(z)} \]

\[ W(z) = \frac{1}{X(z) \ 1 + a_1 z^{-1} + a_2 z^{-2}} \]

\[ \frac{KY(z)}{W(z)} = 1 + b_1 z^{-1} + b_2 z^{-2} \]
Canonical Form II

Same computation as Canonic Form I but requires 2 buses

Never used!!

\[ H(z) = b_0 \frac{z^2 + b_1 z + b_2}{z^2 + a_1 z + a_2} \]
Pole Sensitivity

\[ T(z) = \frac{z^2 + b_1 z + b_2}{z^2 + a_1 z + a_2} \]

\[ z^2 + a_1 z + a_2 \]

-2 Real Part

Magnitude Squared

\[ e.g. \quad z^2 - 1.9z - 0.98 \]

roots at \( \text{Re} = 0.95, \text{Mag} = 0.99 \)
Pole Sensitivity

\[ x \]

\[ \pm \frac{a_1}{2} \]

\[ \sqrt{a_2} \]
Increase Sample Rate

III Conditioned
Why?

\[ T(z) = \frac{Y(z)}{X(z)} = \frac{z^2 + b_1 z + b_2}{z^2 + a_1 z + a_2} \]

\[ Y(z)(z^2 + a_1 z + a_2) = X(z)(z^2 + b_1 z + b_2) \]

\[ Y(z)(1 + a_1 z^{-1} + a_2 z^{-2}) = X(z)(1 + b_1 z^{-1} + b_2 z^{-2}) \]

\[ Y(z) = X(z)(1 + b_1 z^{-1} + b_2 z^{-2}) - Y(z)(a_1 z^{-1} + a_2 z^{-2}) \]

\[ y(n) = x(n) + b_1 x(n-1) + b_2 x(n-2) - a_1 y(n-1) - a_2 y(n-2) \]

For LP filter, \( a_1 \) is negative.

We therefore have positive feedback!
Filter Structure

2 Multipliers for 2 poles - efficient

positive feedback unstable

negative feedback to stabilize

-2*Real

Mag^2
First Order Sections

- How about using first order only?
- Need complex coefficients
- Very low root sensitivity; same as for FIR
- Quantization errors are orthogonal

Orthogonal Quantization Errors
First Order Structures

Complex Coefficients

4 multipliers for two poles - inefficient

To get conjugate pair poles, take real part of output

Real Coefficients
Comments

• Rule of Thumb - Don’t build recursive filters.
• FIR filters probably don’t require more computation if you embed downsampling.
• Only worthwhile IIR filter is probably the leaky integrator (first order).
• Recursive filters are probably not worthwhile, apart from one notable exception - the allpass filters or microripple filters.