Advantages of IIR filters

- Fewer coefficients for sharp cutoff filters
- Can calculate coefficients for standard filters
- Can apply transformations to frequency axis
- Can make all-pass filters (if $b_i = a_{N-i}$)

Disadvantages

- Non-linear phase response
- Can be unstable: adaptive filters difficult, coefficient precision vital

IIR Filter Sharpness

- 21st order FIR, 54th order FIR and 6th order IIR
- IIR filter has steepest cutoff but very non-linear phase
  - Linear phase ($\phi = -\omega \tau$) $\Rightarrow$ pure delay of $\tau$ $\Rightarrow$ Symmetrical FIR
  - The 180º phase jumps arise when the response changes sign

Standard Filters

- Standard filters have specifiable ripple in passband and/or stopband:

Frequency Transformations

- If we replace $z$ by $\frac{z - \alpha}{1 - \alpha z}$ we apply a non-linear frequency transformation.
- Choose $\alpha = \frac{\sin((f_1 - f_2)T\pi)}{\sin((f_1 + f_2)T\pi)}$ to map $f_1$ to $f_2$
- Similar transformations map lowpass to highpass, bandpass or bandstop
Filter Implementation

Transfer Function:

\[ H(z) = \frac{b_0 + b_1 z^{-1} + \cdots + b_N z^{-N}}{1 + a_1 z^{-1} + \cdots + a_M z^{-M}} \]

Recurrence Relation:

\[ y(n) = b_0 x(n) + b_1 x(n-1) + \cdots + b_N x(n-N) \]

\[ -a_1 y(n-1) - \cdots - a_M y(n-M) \]

- Note the sign change of denominator coefficients
- We can factorize H(z) and implement as cascaded filters:

\[ H(z) = \frac{b_0}{1 + a_1 z^{-1} + \cdots + a_M z^{-M}} \]

Filter Implementation

Direct Form I
- FIR followed by IIR
- D represents a one-sample delay (multiplication by \( z^{-1} \))
- We can merge the adders

Direct Form II
- IIR followed by FIR
- Only needs a single delay line

Transpose Networks

- The transfer function is unchanged if you:
  - Reverse the flow in each branch
  - Interchange branch divisions and branch summations
- Coding can be slightly more efficient

Biquad Implementation

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

- We can always factorize the numerator and denominator of \( H(z) \) into quadratic factors with real coefficients.
- Each complex pole pair or zero pair forms a single factor.
- Group nearby pole and zero pairs to form 2\textsuperscript{nd} order sections known as biquads.
- 2\textsuperscript{nd} order filters are much less sensitive to coefficient errors.

C Code for IIR Filters

```c
for (k=order; k>0; k--) {
    w[k] = w[k-1];  v[k] = v[k-1];
    yout += w[k]*b[k]-v[k]*a[k];
}
```

- Software implementation of Direct Form I
- The diagram illustrates the case: order = 3
- We access the delay lines in reverse order so that we can update their values in the second line of the loop.
- \( w[1] \) and \( v[1] \) are updated with junk in the final iteration, then overwritten with their correct values outside the loop.
**Coefficient File**

float a[ ] = { 1, -1.176, 1.1829, -0.2781 };  
float b[ ] = { 0.0181, 0.0543, 0.0543, 0.0181 };  

- Get MATLAB to create a file: coef.txt  
  - See the fopen(), fprintf() and fclose() commands  
- The comma after the last value in each array is optional  
- The length of each array is determined by the number of values specified.  
- In C, arrays go from a[0] ... a[order] whereas in MATLAB, they go from a(1) ... a(order+1).

**Initialisation**

int order;  
float *w, *v;  
#include "coef.txt"  

order = sizeof(a)/sizeof(a[0]) - 1;  
w = (float *) calloc(order+1, sizeof(float));  
v = (float *) calloc(order+1, sizeof(float));  

- To get the number of values the a[ ] array, we divided the length (in bytes) of the whole array by the length of each element.  
- The filter order is one less than the number of a[ ] values.  
- Array names like w and v are pointers (i.e. they just store a memory address).  
- Use calloc() to reserve an area of memory, initialize its contents to 0 and to set w to its starting address.  
- Our code needs the length of w and v to be one greater than the order.

**Circular Buffer**

- In the diagram, M is the value of m as we enter the loop.  
- With ++m, we increment in each time through the loop and ensure that it never exceeds order-1.  
- The final line overwrites the oldest values in the delay line and then decrements m so that they will act as w[M+1] and v[M+1] next time.  
- The w and v arrays only need to contain order elements.

**Summary**

- **IIR filters**  
  - Need fewer coefficients for sharp cut-off filters  
  - Poles and zeros of standard filters can be calculated directly  
  - z-domain transformations can change the cut-off frequency and transform lowpass into highpass or, with a doubling of the order, into bandpass or bandstop.  
- **Alternative Signal-flow Graphs**  
  - Important differences in numerical performance when fixed point arithmetic is used - less important with floating point.  
  - Factorize into biquads to avoid coefficient precision problems.  
- **C implementation**  
  - Use #include directive to insert a coefficient file into program.  
  - Can use circular buffer to implement an efficient delay line.