Today: IIR filter implementation (ctd)
DFT and FFT
Spectral leakage
FFT scaling

Announcements: Fri Oct 3 office hours cancelled
Extra office hours today 12-1:30PM
Deadline for parts orders is Fri Oct 10
Hwk 5 due on thurs Oct 16
Midterm exam on thurs Oct 23.
Coverage: hwks 1-5, labs 1-6, lectures 1-12.

Please keep the lab clean and organized.

Last one out should close the lab door!!!!

In mathematics you don’t understand things, you just get used to them.
— John von Neumann
IIR Canonical direct form 2

a) Non-canonical Direct Form 2.

b) DF2 in canonical form.
Implementing a biquad cascade IIR filter

The implementation steps are:

- Factor the transfer function into pole and zero pairs.
- Choose a biquad architecture, e.g., DF2, TDF2.
- Relate the biquad coefficients to the chosen architecture coefficients.
- Order the poles and the zeros to control internal resonance levels.
- Distribute the gain between the biquad sections.
- Normalize biquad coefficients if necessary.
- Program and get to work.
- Test.
Pole - zero pairing example
Pole - zero pairing example
The above block diagram shows a cascade of four DF2 second order biquad sections. This can be used to implement an eighth order lowpass filter.
The above block diagram shows a cascade of four TDF2 second order biquad sections. This can be used to implement an eighth order lowpass filter.
The DF2 and TDF2 biquad sections

(a) Direct form type 2 biquad section. (b) Transposed direct form 2 biquad section.

\[ H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} \]

The most commonly used biquad is direct form 2. We need to analyze the transfer function magnitudes between input and internal states in addition to between input and output.
Write $H_i = \frac{Y_i}{X_i}$, $H_{i1} = \frac{W_{i1}}{X_i}$ and $H_{i2} = \frac{W_{i2}}{X_i}$. Because of our choice of the $L_\infty$ norm we are interested in the magnitudes of the input to delay stage filter functions:

- **section 1** $H_{11}$ $H_1$
- **section 2** $H_1H_{21}$ $H_1H_2$
- **section 3** $H_1H_2H_{31}$ $H_1H_2H_3$
- **section 4** $H_1H_2H_3H_{41}$ $H_1H_2H_3H_4$
Write $H_i = \frac{Y_i}{X_i}$, $H_{i1} = \frac{W_{i1}}{X_i}$ and $H_{i2} = \frac{W_{i2}}{X_i}$. Because of our choice of the $L_\infty$ norm we are interested in the magnitudes of the input to delay stage filter functions:

- **section 1** $H_{11}$ $H_{12}$ $H_1$
- **section 2** $H_1 H_{21}$ $H_1 H_{22}$ $H_1 H_2$
- **section 3** $H_1 H_2 H_{31}$ $H_1 H_2 H_{32}$ $H_1 H_2 H_3$
- **section 4** $H_1 H_2 H_3 H_{41}$ $H_1 H_2 H_3 H_{42}$ $H_1 H_2 H_3 H_4$
By nominally scaling the input by 4 we can avoid overflow in this realization. If a 12-bit converter is being used and a 16-bit word size, this is no great loss.

NB: this transfer function isn’t the one used in lab.
Biquad input-to-delay stage TFs: TDF2

\[ |H_{11}(f)| \]

\[ |H_{12}(f)| \]

\[ |H_{21}(f)| \]

\[ |H_{22}(f)| \]
Biquad input-to-delay stage TFs: TDF2 (ctd)

$|H_{31}(f)|$

$|H_{32}(f)|$

$|H_{41}(f)|$

$|H_{42}(f)|$
Filter input-to-delay stage TFs: TDF2

\[ |H_{11}(f)| \]

\[ |H_{12}(f)| \]

\[ |H_1(f)H_{21}(f)| \]

\[ |H_1(f)H_{22}(f)| \]
Filter input-to-delay stage TFs: TDF2 (ctd)

\[ |H_1(f)H_2(f)H_{31}(f)| \]

\[ |H_1(f)H_2(f)H_{32}(f)| \]

\[ |H_1(f)H_2(f)H_3(f)H_{41}(f)| \]

\[ |H_1(f)H_2(f)H_3(f)H_{42}(f)| \]
Filter input-to-delay stage TFs: TDF2 - max gain

elliptic TDF2 max of internal TF magnitudes
max: 0.929
Overflow issues for biquad coefficients

Consider the biquad

\[ H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}} = \frac{b_0 z^2 + b_1 z + b_2}{z^2 + a_1 z + a_2}. \]

The poles of \( H(z) \) are determined by \( z^2 + a_1 z + a_2 \). Assume a complex valued pole pair, \( p_1 = r e^{j\theta} \) and \( p_2 = r e^{-j\theta} \).

\[(z - p_1)(z - p_2) = z^2 - 2r \cos(\theta)z + r^2 = z^2 + a_1 z + a_2.\]

In order for the filter to be (conditionally) stable the biquad poles have to be (on or) within the unit circle.

Because \( 0 \leq r \leq 1 \) we have that \( 0 \leq a_2 \leq 1 \) and \( -2 < a_1 \leq 2 \). In addition, \( a_2 \geq a_1^2/4 \).
Overflow issues for biquad coefficients

We will be using Q15 numeric format values in the C5515 and DE2-70. The magnitude of the $a_1$ value can be greater than 1 (but less than 2). We need to worry about this.

There also may be scaling concerns with the $b$ coefficient values as well. One needs to stay alert.

Occasionally there are $b_i$ values with magnitude greater than 1. Large $b$ values can be handled by scaling all of the $b$ coefficients. This affects only the gain through the system.

Scaling cannot be applied to the $a$ values without changing the shape of the transfer function. (Recall $a_0 = 1$ requirement). Alternatives:

1. Implement each multiply $a_i x[n]$ as $((a_i/2)x[n])^2$.
2. Use $Q(14)$ representation.
If we divide the $a$’s by $k$ we need to multiply the sum by $k$.
Use Q14 data and Q14 coefficients?
Q14×Q14 gives Q28. To make Q28 into Q14 shift left 2 then truncate.

Use Q15 data and Q14 coefficients?
Q15×Q14 gives Q29. To make Q29 into Q15 shift left 2 and truncate.

Use Q15 data and Q15 coefficients?
Q15×Q15 gives Q30. To make Q30 into Q15 shift left 1 and truncate.
Need to remember to round before truncating.
Do you need to normalize any coefficients?
Utility of DFT and FFT

▸ Spectrum analysis:
  ▸ Measure frequency response of an analog or digital filter
  ▸ Measure frequency content of a signal (speech, audio, rf, etc)
  ▸ Estimate magnitude or phase of an unknown channel.

▸ Spectrum synthesis: vocoder, voice synthesis, voice scrambler, voice coding, audio effects.

▸ Implementing filters in frequency domain with DFT or FFT:
  ▸ An FIR LTI filter can be implemented by convolving input with impulse response $h[n]$

$$y[n] = \sum_{k=0}^{N-1} h[k] x[n - k] ,$$

▸ ... or by IDFT of the product of $H(k) = \text{DFT}_k (h[n])$ and $\text{DFT}_k (x[n])$

$$Y[k] = H[k] X[k] , \quad k = 0, \ldots, N - 1$$
Filter implementation in time or in frequency

\[ H[k] = \text{DFT}_k(h[n]) = \sum_{n=0}^{N-1} h[n] e^{-j2\pi \frac{k}{N} n} \]

\[ x[n] \xrightarrow{} h[n] \xrightarrow{} y[n] \]

\[ x[n] \xrightarrow{\text{DFT}} H[k] \xrightarrow{\text{IDFT}} y[n] \]
Available: \( N \) time samples \( x[0], \ldots, x[N - 1] \).

DFT is defined for \( k = 0, \ldots, N - 1 \)

\[
X_{DFT}(k) = \text{DFT}_k(x[n]) = \sum_{n=0}^{N-1} x[n] e^{-j2\pi \frac{k}{N} n}
\]

IDFT recovers time samples from DFT

\[
x[n] = \text{IDFT}_n(X_{DFT}(k)) = \frac{1}{N} \sum_{n=0}^{N-1} X_{DFT}(k) e^{j2\pi \frac{k}{N} n}
\]

Rectangular form of DFT

\[
X_{DFT}[k] = \sum_{n=0}^{N-1} x[n] \cos(2\pi nk/N) - j \sum_{n=0}^{N-1} x[n] \sin(2\pi nk/N)
\]
64 point DFT of a discrete time signal
Some properties of DFT

\[ X[k] = \text{DFT}_k(x[n]), \ \{x[n]\}_{n=0}^{N-1} \text{ is real valued, } N \text{ is even integer.} \]

- **Conjugate symmetry:** DFT satisfies \( X[N - k] = X^*[k] \)

- **Magnitude symmetry:**

\[
|X[N/2 + k]| = |X[N/2 - k]|, \quad k = 0, \ldots, N/2
\]

\[
|X[N - k]| = |X[k]|, \quad k = 0, \ldots, N/2
\]

- **Phase anti-symmetry:**

\[
\arg(X[N/2 + k]) = -\arg(X[N/2 - k]), \quad k = 0, \ldots, N/2
\]

\[
\arg(X[N - k]) = -\arg(X[k]), \quad k = 0, \ldots, N/2
\]

\[
\left(\arg X = \text{angle}(X) = \text{atan} \left( \frac{\text{Im}(X)}{\text{Re}(X)} \right) \right)
\]
Q. How many MAC’s does the DFT consume?

\[ X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N - 1. \]

A. The nominal computational cost is \( N^2 \) complex multiply-adds.

Any algorithm that significantly reduces this number can be considered as being fast.

There are many fast DFT algorithms. Some algorithms are faster than others under different circumstances.

We will only cover the original Cooley-Tukey FFT.
Concepts important to FFT

Roots of unity, powers of $W_N = e^{-j2\pi/N}$.

Symmetry of the sine and cosine.

Index mappings.
The decimation-in-time radix-2 FFT

- $N$ is assumed to be an integer power of 2.
- Divide $\{x[n]\}_{n=0}^{N-1}$ into $\{x[2n]\}_{n=0}^{N/2-1}$ and $\{x[2n + 1]\}_{n=0}^{N/2-1}$
- Form the DFT of each set and combine results to form $N$ value DFT.
- Repeat the procedure on each of the $N/2$-point DFTs.
- And so on.

The resulting nominal complex mult-add count is $N \times \log_2(N)$

<table>
<thead>
<tr>
<th>$N$</th>
<th>$\log_2(N)$</th>
<th>$N \times \log_2(N)$</th>
<th>$N^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>3</td>
<td>24</td>
<td>64</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>64</td>
<td>256</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>160</td>
<td>1024</td>
</tr>
<tr>
<td>64</td>
<td>6</td>
<td>384</td>
<td>4096</td>
</tr>
<tr>
<td>128</td>
<td>7</td>
<td>896</td>
<td>16384</td>
</tr>
<tr>
<td>256</td>
<td>8</td>
<td>2048</td>
<td>65536</td>
</tr>
<tr>
<td>512</td>
<td>9</td>
<td>4068</td>
<td>262144</td>
</tr>
<tr>
<td>1024</td>
<td>10</td>
<td>10249</td>
<td>1048576</td>
</tr>
</tbody>
</table>
Easy when you see how it’s done

Start with the forward transform equation

\[ X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N - 1. \]

Express even \( n \) as \( 2p \) and odd \( n \) as \( 2q + 1 \)

\[
X[k] = \sum_{p=0}^{N/2-1} x[2p] e^{-j2\pi k(2p)/N} + \sum_{q=0}^{N/2-1} x[2q + 1] e^{-j2\pi k(2q+1)/N}
\]

\[ = \sum_{p=0}^{N/2-1} x[2p] e^{-j2\pi kp/(N/2)} + e^{-j2\pi k/N} \sum_{q=0}^{N/2-1} x[2q + 1] e^{-j2\pi kq/(N/2)}. \]

Symmetry relation for \( k = 1, \ldots, N/2 - 1 \):

\[
X[k + N/2] = \sum_{p=0}^{N/2-1} x[2p] e^{-j2\pi kp/(N/2)} - e^{-j2\pi k/N} \sum_{q=0}^{N/2-1} x[2q + 1] e^{-j2\pi kq/(N/2)}. \]

Have saved half the computations \((N/2)\). Repeat the process \( \log_2 N \) times.
Example: 8-point radix-2 DIT FFT

DFT MACS: \( N^2 = 64 \)
FFT MACS: \( N \log(N) = 24 \)

\( W_N \) is defined as \( e^{-j2\pi/N} \).

\( N = 8 \) for this example.

Arrows indicate multiplication.

Nodes represent summing.
Example: subdivide again

Multiplication by -1 is the result of a 2-point DFT. Values flow left to right.
Modified 8-point radix-2 FFT diagram

$W$ is defined as $e^{-j2\pi/N}$.

$N = 8$ for this example.

Arrows indicate multiplication.

Nodes represent summing.

Multiplication by -1 is trivial.

Values flow left to right.
Reordering of DIT FFT input/output values

Can reorder the flow graph so that the input is in normal order and the output is in bit reverse order.
Using FFT for signal analysis

- The $k$-th coefficient of the $N$-point FFT of $x[n]$ is a sample of the DTFT of $x[n]$ at digital frequency $f = k/N$.
- If $x[n]$ are time samples $x(nT_s)$ of a continuous time signal $x(t)$ then DTFT is an approximation to the finite time FT of $x(t)$ over the time window $t \in [0, (N - 1)T_s)$.
- There are several issues that need to be addressed
  - Spectral leakage
  - Spectral resolution
  - Time varying spectra
- To build intuition we start by considering an example: DFT of sinusoidal signal.
DFT of a sinusoid at frequency $f_c = m/N$

DFT of sinusoid $x[n] = \cos(2\pi f_c n + \phi)$?

Assume sinusoidal frequency satisfies $f_c = m/N$ for integer $m \in \{0, \ldots, N/2\}$

Use Euler formula: $\cos(\theta) = (e^{j\theta} + e^{-j\theta})/2$

$$X_{DFT}(k) = \frac{e^{j\phi}}{2} \sum_{n=0}^{N-1} e^{-j2\pi \frac{k-m}{N} n} + \frac{e^{-j\phi}}{2} \sum_{n=0}^{N-1} e^{-j2\pi \frac{k+m}{N} n}$$

$$= \begin{cases} \frac{Ne^{j\phi}}{2}, & k = m \\ \frac{Ne^{-j\phi}}{2}, & k = N - m \end{cases}$$

($\Delta[n]$ is kronecker delta function)
DFT of single sinusoid at arbitrary freq

DFT of sinusoid \( x[n] = \cos(2\pi f_c n + \phi) \)?

Assume sinusoidal frequency does not satisfy \( f_c = m/N \) for integer \( m \in \{0, \ldots, N/2\} \)

\[
X_{DFT}(k) = \frac{e^{j\phi}}{2} \sum_{n=0}^{N-1} e^{-j2\pi \frac{k-Nf_c}{N} n} + \frac{e^{-j\phi}}{2} \sum_{n=0}^{N-1} e^{-j2\pi \frac{k+Nf_c}{N} n} \nonumber
\]

This is the leakage phenomenon and it occurs when \( f_c \neq m/N \).
DFT: a sinusoid $f_c = 0.25 = m/N, \ m = 4, \ N = 64$
DFT: single sinusoid $f_c = 4.5/N$, not an integer/N
DFT: two orthogonal sinusoids $f_{ci} = m_i/N,$
$m_1 = 4, \ m_2 = 5, \ N = 64$
DFT: two non-orthogonal sinusoids

\[ f_{ci} = \frac{m_i}{N}, \quad m_1 = 4, \quad m_2 = 4.5, \quad N = 64 \]
Two orthogonal sinusoids $f_{ci} = m_i/N$, $m_1 = 4$, $m_2 = 5$, $N = 64$
Two nonorthogonal sinusoids $f_{ci} = m_i/N$, $m_1 = 4$, $m_2 = 4.5$, $N = 64$
FFT input scaling

Consider using standard 16-bit Q15 number representation in FFT as in AIC3204.

Let input to the FFT be the cosine signal

\[ \cos(2\pi f_c n) = \frac{e^{j2\pi f_c n} + e^{-j2\pi f_c n}}{2}, \quad n = 0, \ldots, N - 1 \]

Overflow problem 1: The gain at the \( f_c \) frequency (assuming it matches some analysis frequency \( m/N \)) is \( N/2 \). If a 1024 point transform is taken then the result might require 10-1+16 = 25 bits.

Overflow problem 2: A complex input with 16 bit Q15 real and imaginary parts can overflow if a phase rotation occurs. For example, \( 1 + j1 \) can rotate to \( 1.414 + j0 \) creating an overflow in the Q15 real part.

This is why in lab 6 you will be implementing 32 bit precision FFT’s.
An approach to scaling

- **Normalization**

  Consider a Q15 sinewave input having amplitude 1. Using $1/N$ scaling on the forward transform, the magnitude of the FFT output will be capped at $1/2$.

- **Distribute normalization over each of the $\log_2(N)$ layers of FFT**

  Assume $N = 2^n$ is a power of two, $n = \log_2 N$ an integer. Then can apply a scale factor of $1/2$ to each layer of the FFT. The net effect will be to scale the FFT operation by $1/N$. 
Summary of what we covered today

- The DFT and FFT
- Finite precision and scaling issues for FFT
- Spectral leakage
References

