

15. Digital Filters

Chapter 15 objectives are to:

- Introduce basic principles involved in digital filtering,
- Define the Z Transform and use it to analyze filters,,
- Develop digital filter implementations
- Introduce DFT and some applications

15.1. Basic Principles

$x_c(t)$ is a continuous analog signal. f_s is the sample rate

$$x(n) = x_c(nT) \quad \text{with } -\infty < n < +\infty. \tag{1}$$

There are two types of approximations associated with the sampling process.

- finite precision of the ADC
- finite sampling frequency.

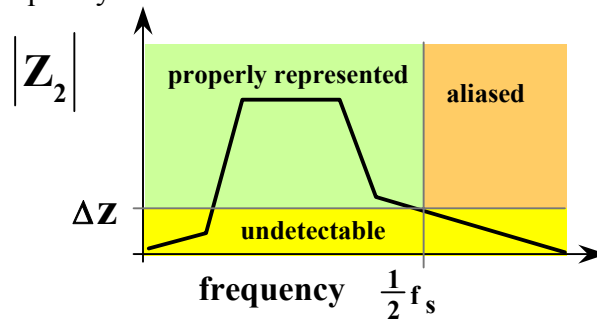


Figure 12.30. To prevent aliasing there should be no measurable signal above $0.5f_s$.

A **causal** digital filter calculates

$y(n)$ from $y(n-1)$, $y(n-2)$,... and $x(n)$, $x(n-1)$, $x(n-2)$,...
not future data (e.g., $y(n+1)$, $x(n+1)$ etc.)

A **linear** filter is constructed from a linear equation.

A **nonlinear** filter is constructed from a nonlinear equation.

One nonlinear filter is the median.

A **finite impulse response** filter (FIR) relates $y(n)$ only in terms of $x(n)$, $x(n-1)$, $x(n-2)$,...

$$y(n) = \frac{x(n) + x(n-3)}{2} \tag{4}$$

An **infinite impulse response** filter (IIR) relates $y(n)$ in terms of both $x(n)$, $x(n-1)$,..., and $y(n-1)$, $y(n-2)$,...

$$y(n) = \frac{113 \bullet x(n) + 113 \bullet x(n-2) - 97 \bullet y(n-2)}{128}$$

The definition of the Z-Transform:

$$X(z) = Z[x(n)] \equiv \sum_{n=-\infty}^{\infty} x(n) z^{-n} \tag{6}$$

Consider the Laplace Transform

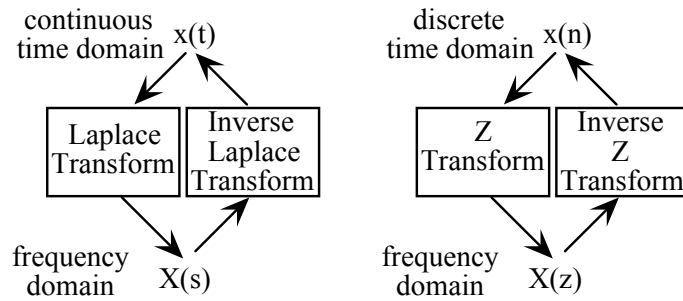


Figure 15.1. A transform is used to study a signal in the frequency domain.

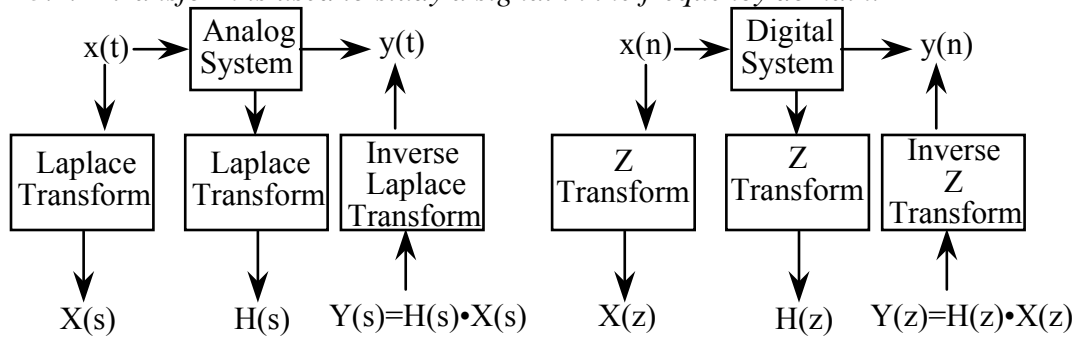


Figure 15.2. A transform can also be used to study a system in the frequency domain.

The gain = $|H(s)|$ at $s = j 2\pi f$, for all frequencies, f .

The phase = $\text{angle}(H(s))$ at $s = j 2\pi f$.

The gain and phase of a digital system is specified in its transform, $H(z) = Y(z)/X(z)$.
from DC to $\frac{1}{2} f_s$

One can use the definition of the Z-Transform to prove that:

$$Z[x(n-m)] = z^{-m} Z[x(n)] = z^{-m} X(z) \tag{7}$$

For example if $X(z)$ is the Z-Transform of $x(n)$,

then $z^{-2} \cdot X(z)$ is the Z-Transform of $x(n-2)$.

$$H(z) \equiv \frac{Y(z)}{X(z)} \tag{8}$$

To find the response of the filter, let z be a complex number on the unit circle

$$z \equiv e^{j2\pi f/f_s} \quad \text{for } 0 \leq f < \frac{1}{2} f_s \tag{9}$$

or

$$z = \cos(2\pi f/f_s) + j \sin(2\pi f/f_s) \tag{10}$$

Let

$$H(f) = a + bj \quad \text{where } a \text{ and } b \text{ are real numbers}$$

The gain of the filter is the complex magnitude of $\mathbf{H}(z)$ as \mathbf{f} varies from 0 to $\frac{1}{2} f_s$.

$$\text{Gain} \equiv |\mathbf{H}(f)| = \sqrt{a^2 + b^2} \quad (12)$$

The phase response of the filter is the angle of $\mathbf{H}(z)$ as \mathbf{f} varies from 0 to $\frac{1}{2} f_s$.

$$\text{Phase} \equiv \text{angle}[\mathbf{H}(f)] = \tan^{-1} \frac{b}{a} \quad (13)$$

15.2. Simple Digital Filter Examples.

Although this filter appears to be simple, we can use it to implement a low-Q 60 Hz notch.

$$y(n) = \frac{x(n) + x(n-3)}{2} \quad (4)$$

Again we take the Z-Transform of both sides of Equation 4:

$$Y(z) = \frac{X(z) + z^{-3} X(z)}{2} \quad (24)$$

Next we rewrite the equation in the form of $\mathbf{H}(z)=Y(z)/X(z)$.

$$\mathbf{H}(z) \equiv \frac{Y(z)}{X(z)} = \frac{1 + z^{-3}}{2} \quad (25)$$

We then can use Equations 9 through 13 to determine the gain and phase response of this filter.

$$\mathbf{H}(f) = \frac{1 + e^{-j6\pi f/f_s}}{2} = \frac{1 + \cos(6\pi f/f_s) - j \sin(6\pi f/f_s)}{2}$$

$$\text{Gain} \equiv |\mathbf{H}(f)| = 0.5 \sqrt{\{1 + \cos(6\pi f/f_s)\}^2 + \{\sin(6\pi f/f_s)\}^2}$$

```
// fs=360Hz, channel specifies ADC channel
unsigned short x[4],y;
// x[0] is x(n) current sample
// x[1] is x(n-1) sample 1/360 sec ago
// x[2] is x(n-2) sample 2/360 sec ago
// x[3] is x(n-3) sample 3/360 sec ago
#define C5F 0x20
// period 1500000/360
interrupt 13 void TOC5handler(void){
    TFLG1 = OC5;          // ack C5F
    TC5 = TC5+4167;      // fs=360Hz
    x[3] = x[2];         // shift MACQ data
    x[2] = x[1];
    x[1] = x[0];
    x[0] = ADC_In(channel); // new data
    y = (x[0]+x[3])>>1;
    PutFifo(y);}
void ritual(void) {
    asm sei              // make atomic
    TIOS |= C5F;        // enable C5F
    TSCR1 = 0x80;      // Enable TCNT
    TSCR2 = 0x04;      // 667ns, TOI disarm
    TIE |= C5F;        // Arm output compare 5
```

```

x[0]=x[1]=x[2]=x[3]=0;
TFLG1=OC5; // Initially clear C5F
TC5 = TCNT+4167;
asm cli
}

```

Program 15.3. Real time data acquisition with a simple digital filter, Equation 4.

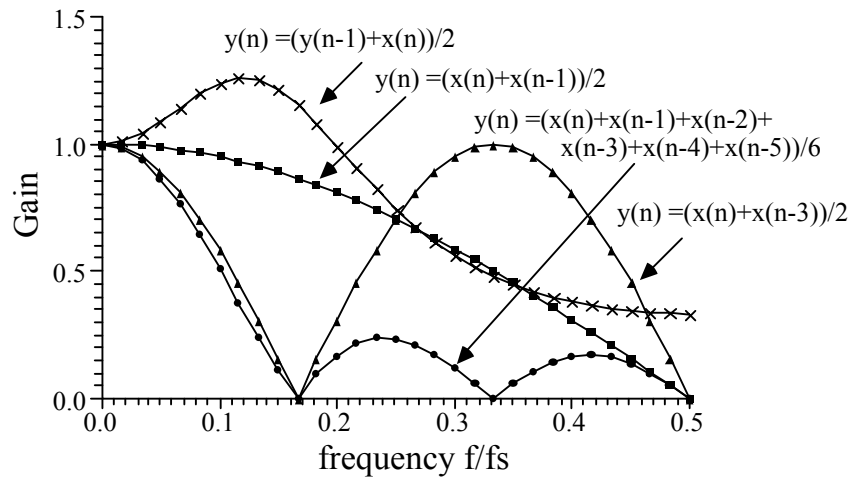


Figure 15.3. Gain versus frequency response for four simple digital filters.

15.4. High-Q 60 Hz Digital Notch Filter

There are two objectives for this example
 show an example of a digital notch filter,
 demonstrate the use of fixed-point math.

60 Hz noise is a significant problem in most data acquisition systems. The 60 Hz noise reduction can be accomplished:

- 1) Reducing the noise source, e.g., shut off large motors;
- 2) Shielding the transducer, cables, and instrument;
- 3) Implement a 60 Hz analog notch filter;
- 4) Implement a 60 Hz digital notch filter.

analog condition	digital condition	consequence
zero near $s=j2\pi f$ line	zero near $z=e^{j2\pi f/fs}$	low gain near the zero
pole near $s=j2\pi f$ line	pole near $z=e^{j2\pi f/fs}$	high gain near the pole
zeros in conjugate pairs	zeros in conjugate pairs	the output $y(t)$ is real
poles in conjugate pairs	poles in conjugate pairs	the output $y(t)$ is real
poles in left half plane	poles inside unit circle	stable system
poles in right half plane	poles outside unit circle	unstable system
pole near a zero	pole near a zero	high Q response

Table 15.1. Analogies between the analog and digital filters.

It is the 60 Hz digital notch filter that will be implemented in this example. The signal is sampled at $f_s=480$ Hz. We wish to place the zeros (gain=0) at 60 Hz, thus

$$\theta = \pm 2\pi \cdot \frac{60}{f_s} = \pm \pi/4 \tag{41}$$

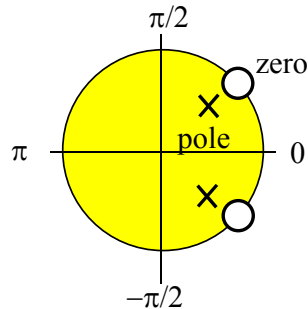


Figure 15.9. Pole-zero plot of a 60 Hz digital notch filter.

The zeros are located on the unit circle at 60 Hz

$$z_1 = \cos(\theta) + j \sin(\theta) \quad z_2 = \cos(\theta) - j \sin(\theta) \tag{42}$$

To implement a flat pass band away from 60 Hz the poles are placed next to the zeros, just inside the unit circle. Let α define the closeness of the poles where $0 < \alpha < 1$.

$$p_1 = \alpha z_1 \quad p_2 = \alpha z_2 \tag{43}$$

for $\alpha = 0.75$

The transfer function is

$$H(z) = \prod_{i=1}^k \frac{(z - z_i)}{(z - p_i)} = \frac{(z - z_1)(z - z_2)}{(z - p_1)(z - p_2)} \tag{45}$$

which can be put in standard form (i.e., with terms $1, z^{-1}, z^{-2} \dots$)

$$H(z) = \frac{1 - 2 \cos(\theta)z^{-1} + z^{-2}}{1 - 2\alpha \cos(\theta)z^{-1} + \alpha^2 z^{-2}} \tag{46}$$

or

$$H(z) = \frac{1 - \sqrt{2}z^{-1} + z^{-2}}{1 - \frac{3}{4}\sqrt{2}z^{-1} + \frac{9}{16}z^{-2}} \tag{47}$$

The digital filter can be derived by taking the inverse Z-transform of the H(z) equation

$$y(n) = x(n) - 2\cos(\theta)x(n-1) + x(n-2) + 2\alpha\cos(\theta)y(n-1) - \alpha^2y(n-2)$$

or

$$y(n) = x(n) - 1.41421x(n-1) + x(n-2) + 1.06066y(n-1) - 0.5625y(n-2)$$

we rewrite using fixed-point math

$$y(n) = (128x(n)-181x(n-1)+128x(n-2)+136y(n-1) -72y(n-2))/128$$

Another consideration for this type of filter is the fact that the gain in the pass bands is greater than one. The first method is to use the $H(z)$ transfer equation and set $z=1$.

$$H(1) = \frac{1 - \sqrt{2} + 1}{1 - \frac{3}{4}\sqrt{2} + \frac{9}{16}} \approx 1.1673 \approx 128/110$$

We can adjust the digital filter so that the DC gain is exactly 1, by prescaling the input terms by 110/128.

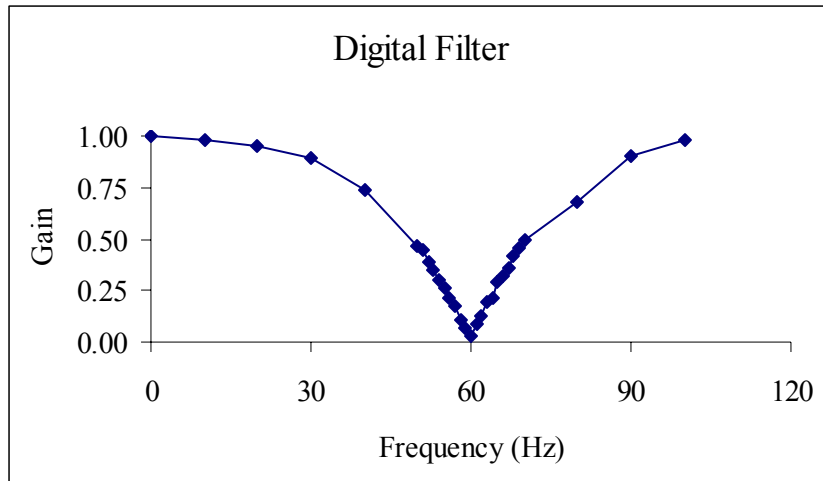
$$y(n) = (110x(n)-156x(n-1)+110x(n-2)+136y(n-1) -72y(n-2))/128$$

*****copy paste filter into dftest10.c*****

```
#define fs 480
unsigned char x[3];
unsigned char y[3];
unsigned char Filter(unsigned char data){
    y[2] = y[1]; y[1] = y[0]; // shift MACQ
    x[2] = x[1]; x[1] = x[0];
    x[0] = data; // new data
    y[0] = (110*(x[0]+x[2])
        -156*x[1]+136*y[1]-72*y[2]+64)/128;
    return y[0];
}
```

```
f = 0 Hz   Gain = 1.00
f = 10 Hz  Gain = 0.98
f = 20 Hz  Gain = 0.95
f = 30 Hz  Gain = 0.89
f = 40 Hz  Gain = 0.74
f = 50 Hz  Gain = 0.47
f = 51 Hz  Gain = 0.45
f = 52 Hz  Gain = 0.39
f = 53 Hz  Gain = 0.35
f = 54 Hz  Gain = 0.30
f = 55 Hz  Gain = 0.26
f = 56 Hz  Gain = 0.21
f = 57 Hz  Gain = 0.17
f = 58 Hz  Gain = 0.11
f = 59 Hz  Gain = 0.07
f = 60 Hz  Gain = 0.03
f = 61 Hz  Gain = 0.09
f = 62 Hz  Gain = 0.13
f = 63 Hz  Gain = 0.19
f = 64 Hz  Gain = 0.21
f = 65 Hz  Gain = 0.29
f = 66 Hz  Gain = 0.32
f = 67 Hz  Gain = 0.36
f = 68 Hz  Gain = 0.42
f = 69 Hz  Gain = 0.46
f = 70 Hz  Gain = 0.50
```

f = 80 Hz Gain = 0.68
 f = 90 Hz Gain = 0.90
 f = 100 Hz Gain = 0.98



The “Q” of a digital notch filter is defined to be

$$Q \equiv \frac{f_c}{\Delta f} \tag{59}$$

where f_c is the center or notch frequency, and Δf frequency range where is gain is below 0.707 of the DC gain.

This is a low $Q = 60/40 = 1.5$

In general, if f_s is $2 \cdot k \cdot f_c$ Hz (where k is any integer $k \geq 2$), then the following is a f_c notch filter:

$$y(n) = \frac{x(n) + x(n-k)}{2} \tag{34}$$

Similarly, if f_s is $k \cdot f_c$ Hz (where k is any integer $k \geq 2$), then the following rejects f_c and its harmonics: $2f_c, 3f_c \dots$

$$y(n) = \frac{1}{k} \sum_{i=0}^{k-1} x(n-i) \tag{35}$$

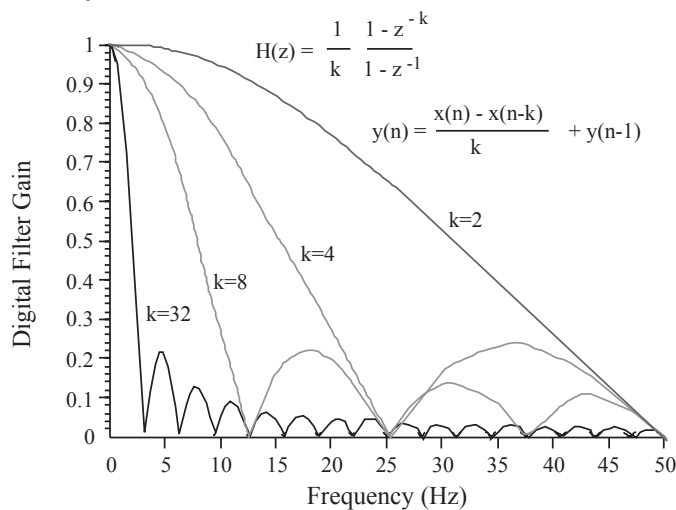


Figure 15.17. Gain versus frequency plot of four low pass filters.

The Discrete Fourier Transform

- The DFT of a block of N time samples $\{a_n\} = \{a_0, a_1, a_2, \dots, a_{N-1}\}$ is a set of N frequency bins $\{A_m\} = \{A_0, A_1, A_2, \dots, A_{N-1}\}$ where:

$$A_m = \sum_{n=0}^{N-1} a_n W_N^{mn} \quad m=0,1,2,\dots,N-1$$

$$W_N \equiv e^{-j2\pi/N}$$

- While the DFT deals only with samples and bins, with no concept of seconds and Hz, when looking at ADC samples spaced at intervals T (in sec)
 - Frequency bin m represents components at $m \cdot f_s / N$ (in Hz)
- The DFT resolution in Hz/bin is the reciprocal of the total time spent gathering time samples; i.e., $1/(NT)$

```

realttype *xr, realtype *yi, int n, int i
realttype magnitude, nr, rr, ii;

nr = (realttype) n; // number of points
if (i == 0)
  magnitude =
    sqrt(xr[0]*xr[0]+yi[0]*yi[0])/nr;

```

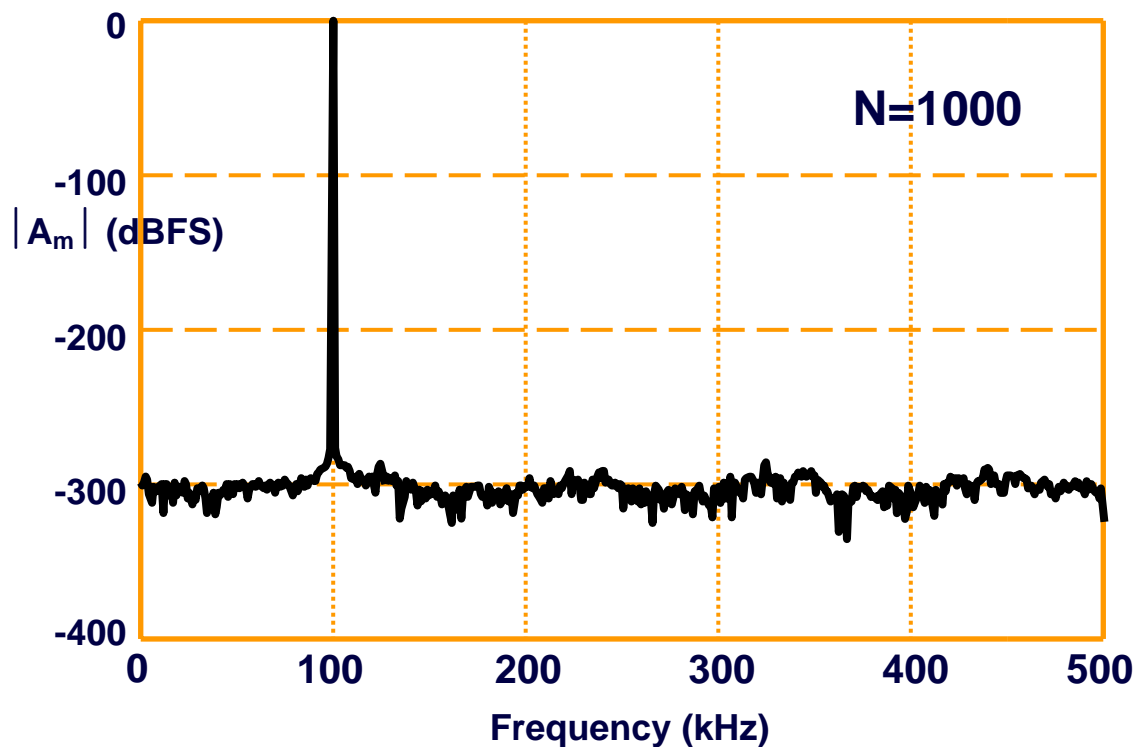
```
else{  
    rr = fabs(xr[i]) + fabs(xr[n-i]);  
    ii = fabs(yi[i]) + fabs(yi[n-i]);  
    magnitude = sqrt(rr*rr+ii*ii)/nr;  
}  
dBFS = 20.0*log10(magnitude/2.5);           // full scale  
is 2.5 volts
```

Applications

Measure S/N ratio

Identify noise

1 Vrms, 100kHz Sinewave DFT



Windowing

- Spectral leakage can be virtually eliminated by “windowing” time samples prior to the DFT
 - Windows taper smoothly down to zero at the beginning and the end of the observation window
 - Time samples are multiplied by window coefficients on a sample-by-sample basis
- Windowing sinewaves places the window spectrum at the sinewave frequency
 - Convolution in frequency