Digital Signal Processing and Filter Design using Scilab

Iman Mukherjee

Department of Electrical Engineering, IIT Bombay

December 1, 2010

Outline

1. Basic signal processing tools
   - Discrete Fourier Transform
   - Fast Fourier Transform
   - Convolution
   - Plotting
   - Group Delay
   - Aliasing
2. Filter Design
   - Non-Recipe Based
   - Recipe Based
   - An Example Application

DFT

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

DFT

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

- The Scilab command --→ [xf] = dft(x,flag);

DFT

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

- The Scilab command --→ [xf] = dft(x,flag);
- x is the time domain representation

DFT

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

- The Scilab command --→ [xf] = dft(x,flag);
- x is the time domain representation
- xf is the frequency domain representation

DFT

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

- The Scilab command --→ [xf] = dft(x,flag);
- x is the time domain representation
- xf is the frequency domain representation
- flag = 1 or -1

DFT

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

- The Scilab command --→ [xf] = dft(x,flag);
- x is the time domain representation
- xf is the frequency domain representation
- flag = 1 or -1
- Notice - Cosine is Even Symmetric, hence this 64-point DFT is real with peaks at 4 and 60 (64-4)

DFT

$$X(\omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega n}$$

- The Scilab command --→ [xf] = dft(x,flag);
- x is the time domain representation
- xf is the frequency domain representation
- flag = 1 or -1
- Notice - Cosine is Even Symmetric, hence this 64-point DFT is real with peaks at 4 and 60 (64-4)
- Faster way - fft ...

FFT

FFT

- x=fft(a ,-1) or x=fft(a)

FFT

- x=fft(a ,-1) or x=fft(a)
- y=fft2(x,n,m) - two-dimension

FFT

- x=fft(a ,-1) or x=fft(a)
- y=fft2(x,n,m) - two-dimension
- x=fft(a,-1,dim,incr) - multidimensional fft

FFT

- x=fft(a ,-1) or x=fft(a)
- y=fft2(x,n,m) - two-dimension
- x=fft(a,-1,dim,incr) - multidimensional fft
- fftshift(abs(y)) - rearranges the fft output, moving the zero frequency to the center of the spectrum

Exercise 1

The convol Command

- With the convol command

The convol Command

- With the convol command
- Without the convol command (multiplying in the frequency domain)

Exercise 2

Bode and Pole-Zero Plots

- Demo Pole-Zero Plot

Bode and Pole-Zero Plots

- Demo Pole-Zero Plot
- Demo Bode Plot

Group Delay

Rate of change of Phase w.r.t Frequency

$$\tau_g = \frac{d\phi}{d\omega}$$

Where,
τ_g is the Group Delay
ϕ is for phase delay
ω is for frequency

What is Aliasing?

- Ambiguity from reconstruction !

What is Aliasing?

- Ambiguity from reconstruction !
- Shannon-Nyquist Sampling theorem.

What is Aliasing?

- Ambiguity from reconstruction !
- Shannon-Nyquist Sampling theorem.
- Under-sampling

What is Aliasing?

- Ambiguity from reconstruction !
- Shannon-Nyquist Sampling theorem.
- Under-sampling

- Scilab commands to remember -
- t = soundsec(n [,rate]) - generates n sampled seconds of time parameter
- v = linspace(x1,x2 [,n]) - linearly spaced vector
- mtlb_hold(flag)

- First order filter

- First order filter
- Second order filter

- First order filter
- Second order filter
- Observation : More attenuation !

- First order filter
- Second order filter
- Observation : More attenuation !

- Scilab commands to remember -
- ss2tf, tf2ss, dscr - State-Space ¡-¿ Transfer Function, Discretizing Continuous Systems
- r = repfreq(Sys,frq) - Frequency response
- playsnd(data)
- filtered_output = flts(input,filter)

Analog filter family prototypes

- Butterworth

Analog filter family prototypes

- Butterworth
- Chebyshev

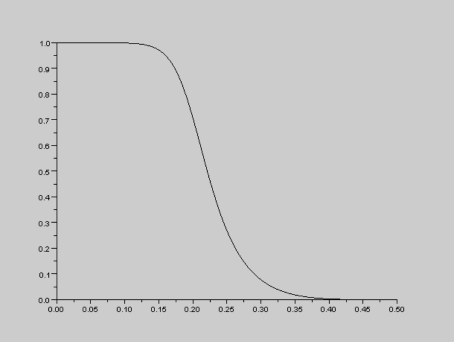
Analog filter family prototypes

- Butterworth
- Chebyshev
- Inverse Chebyshev

Analog filter family prototypes

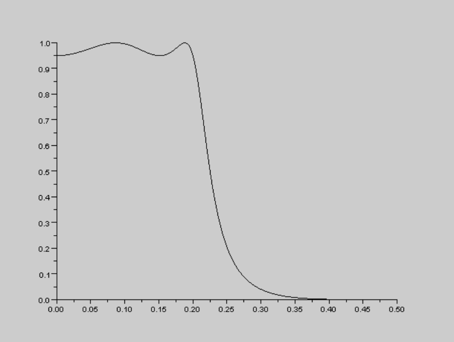
- Butterworth
- Chebyshev
- Inverse Chebyshev
- Elliptic/Chauer

Butterworth



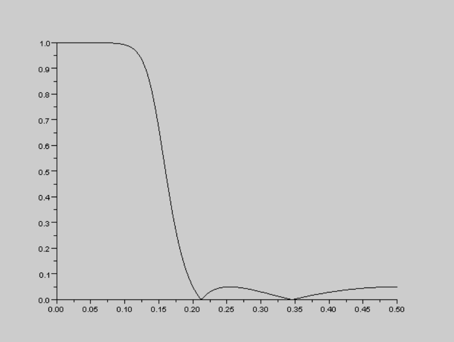
- Passband: Monotonic
- Stopband: Monotonic
- No ripples
- Wide Transition, slow Roll-off

Chebyshev



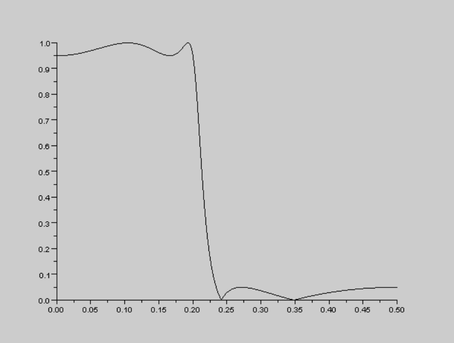
- Passband: Equiripple
- Stopband: Monotonic
- Only ripples in Passband
- Lesser Transition width, slow Roll-off at high frequencies

Inverse Chebyshev



- Passband: Monotonic
- Stopband: Equiripple
- Only ripples in Stopband
- Lesser Transition width, slow Roll-off at low frequencies

Elliptic



- Passband: Equiripple
- Stopband: Equiripple
- Ripples in both - Passband and Stopband
- Least Transition, sharp and fast Roll-off

An example

Steps :

- Decide over the filter specifications (type, PB and SB cut-offs, ripples etc.)

Let us design a Low-Pass Chebyshev Filter

An example

Steps :

- Decide over the filter specifications (type, PB and SB cut-offs, ripples etc.)
- Normalize

Let us design a Low-Pass Chebyshev Filter

An example

Steps :

- Decide over the filter specifications (type, PB and SB cut-offs, ripples etc.)
- Normalize
- Calculate the Order of the filter

Let us design a Low-Pass Chebyshev Filter

An example

Steps :

- Decide over the filter specifications (type, PB and SB cut-offs, ripples etc.)
- Normalize
- Calculate the Order of the filter
- Calculate the filter coeffiecients

Let us design a Low-Pass Chebyshev Filter

An example

Steps :

- Decide over the filter specifications (type, PB and SB cut-offs, ripples etc.)
- Normalize
- Calculate the Order of the filter
- Calculate the filter coeffiecients
- Implement Analog Transfer function

Let us design a Low-Pass Chebyshev Filter

An example

Steps :

- Decide over the filter specifications (type, PB and SB cut-offs, ripples etc.)
- Normalize
- Calculate the Order of the filter
- Calculate the filter coeffiecients
- Implement Analog Transfer function
- Perform bilinear transformation

Let us design a Low-Pass Chebyshev Filter

An example

Steps :

- Decide over the filter specifications (type, PB and SB cut-offs, ripples etc.)
- Normalize
- Calculate the Order of the filter
- Calculate the filter coeffiecients
- Implement Analog Transfer function
- Perform bilinear transformation
- Convert to the needed filter type

Let us design a Low-Pass Chebyshev Filter

Scilab Commands in Filter Design

- iir - [hz]=iir(n,ftype,fdesign,frq,delta)
- eqiir - [cells,fact,zzeros,zpoles]=eqiir(ftype,approx,om,deltap,deltas)
- eqfir - [hn]=eqfir(nf,bedge,des,wate)
- wfir - [wft,wfm,fr]=wfir(ftype,forder,cfreq,wtype,fpar)
- analpf - [hs,pols,zers,gain]=analpf(n,fdesign,rp,omega)
- trans - hzt=trans(pd,zd,gd,tr_type,frq)

Exercise 3

Audio effects

- Flanging - A sound file seems like riding on a wave
- Echo - Delayed and added back
- Equalizer - Different Frequency Bands
  - Low
  - Mid
  - High

Thank you !

- Sanjit K. Mitra, "Digital Dignal Processing, A computer based approach", *Tata McGraw-Hill Edition 1998.*
- Steven W. Smith, "http://www.dspguide.com/ "
- Carey Bunks, Franc ,ois Delebecque, Georges Le Vey and Serge Steer, "Scilab Group INRIA Meta2 Project/ENPC Cergrene ", *Signal Processing with Scilab.*