Summary of Matlab Signal processing Toolbox

Zhuoqi **Cheng**
zch@mmmi.sdu.dk
SDU Robotics

Important for exam

→ Theoretically, you can finish the exam without Matlab. But Matlab is strongly recommanded in the exam.
→ You need to install 'Observer' (not 'exam monitor' anymore).

→ **You have to disable Matlab copilot before the exam!**
→ **Using copilot in the exam, detected by 'Observer', will be considered as cheating.**

→ But you are allowed to use '**help** …' in Matlab to consult the use of function.

Computation

conv(x,h,'same')	convolution multiplication
real()	Get the real part of a number
imag()	Get the imagine part of a number
log10()	Calcultion of ln()
root()	Calculate roots of a polynomial
[r,p,k]=residue(b,a)	Partial fraction decomposition (s)
[r,p,k]=residuez(b,a)	Partial fraction decomposition (z^{-1})
syms s	Symbolic notation
subs(eq, var, num)	Substitute a var in equation by number

Fourier transform & Z transform

fft()	Compute Fast Fourier Transform (FFT)
ifft()	Compute inverse Fast Fourier Transform (IFFT)
fftshift()	Shift zero-frequency component to the center
ifftshift()	Inverse zero-frequency shift
ztrans()	Z transform of a symbolic section
iztrans()	Inverse Z transform of a symbolic section

Spectrum analysis

abs()	Magnitude spetrum
angle()	Phase angle
bode()	Bode plot

Transfer function and response

Hs = tf(b, a)	Create an analog filter (s-domain)
Hz = tf(b, a, Ts)	Create a digital filter (z-domain)
y = impulse(sys,t)	Impulse response
y = freqs(b,a,w)	Frequency response for analog filter
y = freqz(b,a,n)	Frequency response for digital filter
[num,den] = zp2tf(z,p,k)	Convert(z,p,k) to transfer function's numerator and denominator
[z,p,k] = tf2zp(b,a)	From transfer function to find its zeros, poles and gain
[p,z] = pzmap(sys)	Zero-pole plot, also receive the zeros and poles values
pzplot(sys)	Zero-pole plot
y = lsim(sys, x, t)	Simulate an input signal passing through filter
y = filter(b,a,x)	Filtered result with input x and transfer function (b,a)

Second order sections (SOS)

[sos,g] = tf2sos(b,a)

Transfer function coefficients, specified as vectors. Express the transfer function in terms of b and a as

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_1 + b_2 z^{-1} + \cdots + b_{n+1} z^{-n}}{a_1 + a_2 z^{-1} + \cdots + a_{m+1} z^{-m}}.$$

Second-order section representation, returned as a matrix. sos is an L-by-6 matrix

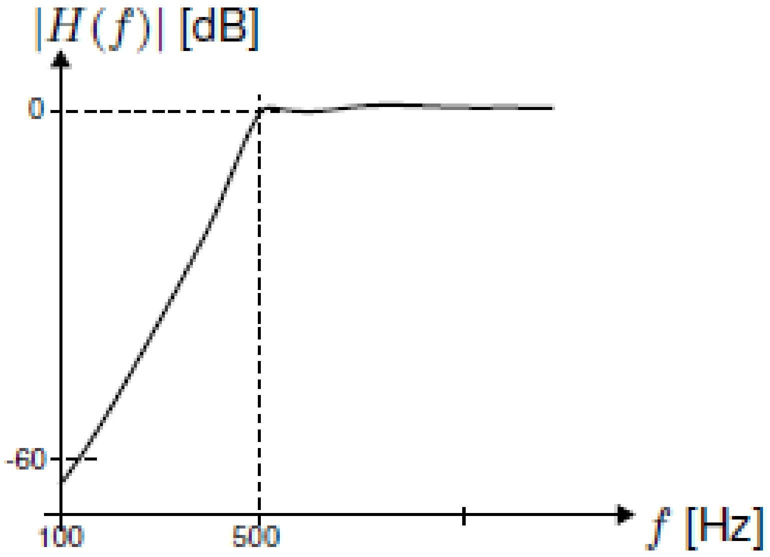
$$\text{sos} = \begin{bmatrix} b_{01} & b_{11} & b_{21} & 1 & a_{11} & a_{21} \\ b_{02} & b_{12} & b_{22} & 1 & a_{12} & a_{22} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ b_{0L} & b_{1L} & b_{2L} & 1 & a_{1L} & a_{2L} \end{bmatrix}$$

whose rows contain the numerator and denominator coefficients b_{ik} and a_{ik} of the second-order sections of $H(z)$:

$$H(z) = g \prod_{k=1}^{L} H_k(z) = g \prod_{k=1}^{L} \frac{b_{0k} + b_{1k} z^{-1} + b_{2k} z^{-2}}{1 + a_{1k} z^{-1} + a_{2k} z^{-2}}.$$

Example: Design of high pass filter

Design a 0.5 dB Chebyshev highpass filter with cutoff frequency 500Hz, sampling frequency 8kHz.




```
% construct analog filter
[b,a] = cheby1(4,0.5,2*pi*500,'high',"s");
Hs=tf(b,a);

% convert it to digital filter (bilinear)
Hz = c2d(Hs, 1/8000,'tustin');
bode(Hz)

% convert to second order sections
bb =cell2mat(Hz.Numerator);
aa = cell2mat(Hz.Denominator);
[ss,g] = tf2sos(bb,aa);
Hz1 = tf(ss(1,1:3), ss(1,4:6), 1/8000);
Hz2 = tf(ss(2,1:3), ss(2,4:6), 1/8000);
Hz_csc = g*Hz1*Hz2;
bode(Hz_csc)
```

Analog filter design

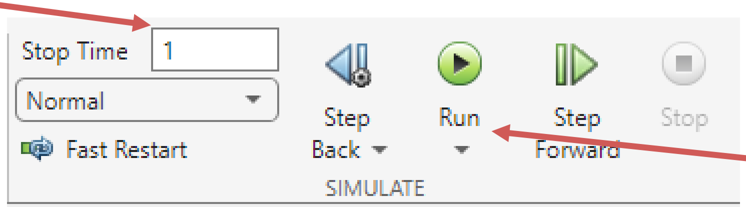
[b,a] = butter(N, wa, ‘s’)	Analog Butterworth filter
[b,a] = cheby1(N, Rp, wa, ‘s’)	Analog Chebyshev type I filter
[b,a] = besself(N, wa)	Analog Bessel filter
[z,p,k] = buttap(n)	Butterworth frequency-normalized filter
[z,p,k] = cheb1ap(n,Rp)	Chebyshev type I frequency-normalized filter
[z,p,k] = besselap(n)	Bessel frequency-normalized filter
[bt, at] = lp2lp(b,a,Wo)	Transform lowpass analog filters to lowpass
[bt, at] = lp2hp(b,a,Wo)	Transform lowpass analog filters to highpass
[bt, at] = lp2bp(b,a,Wo,Bw)	Transform lowpass analog filters to bandpass
[bt, at] = lp2bs(b,a,Wo,Bw)	Transform lowpass analog filters to bandstop

Digital Filter

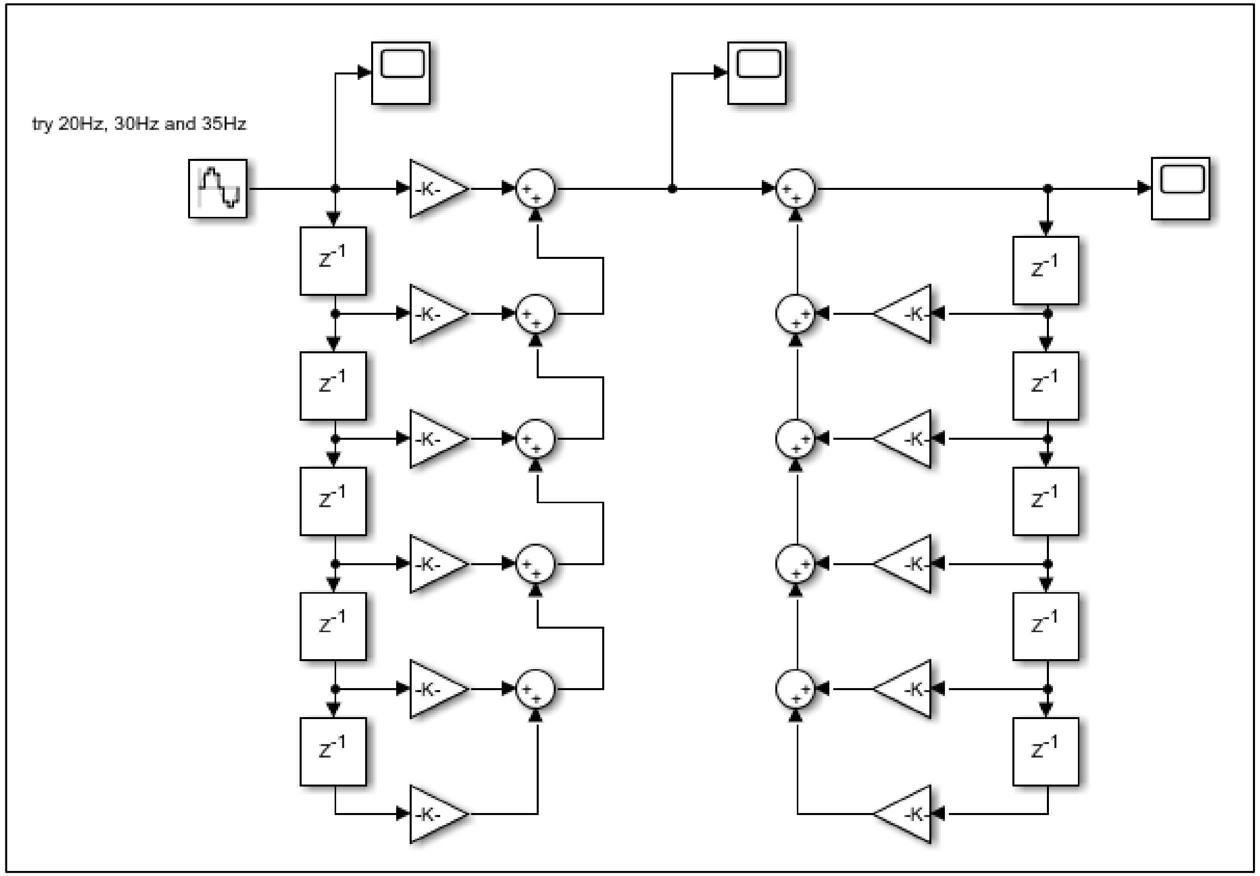
fir1(n,Wn,ftype,window)	Generate a n_order FIR filter with window
barlett(L)	Barlett window length L
hamming(L)	Hamming window length L
hanning(L)	Hanning window length L
kaiser(L, beta)	Kaiser window length L
Hz = c2d(Hs, Ts, method)	Convert analog filter Hs to digital filter Hz (method: ‘matched’, ‘impulse’, ‘tustin’)

Realization

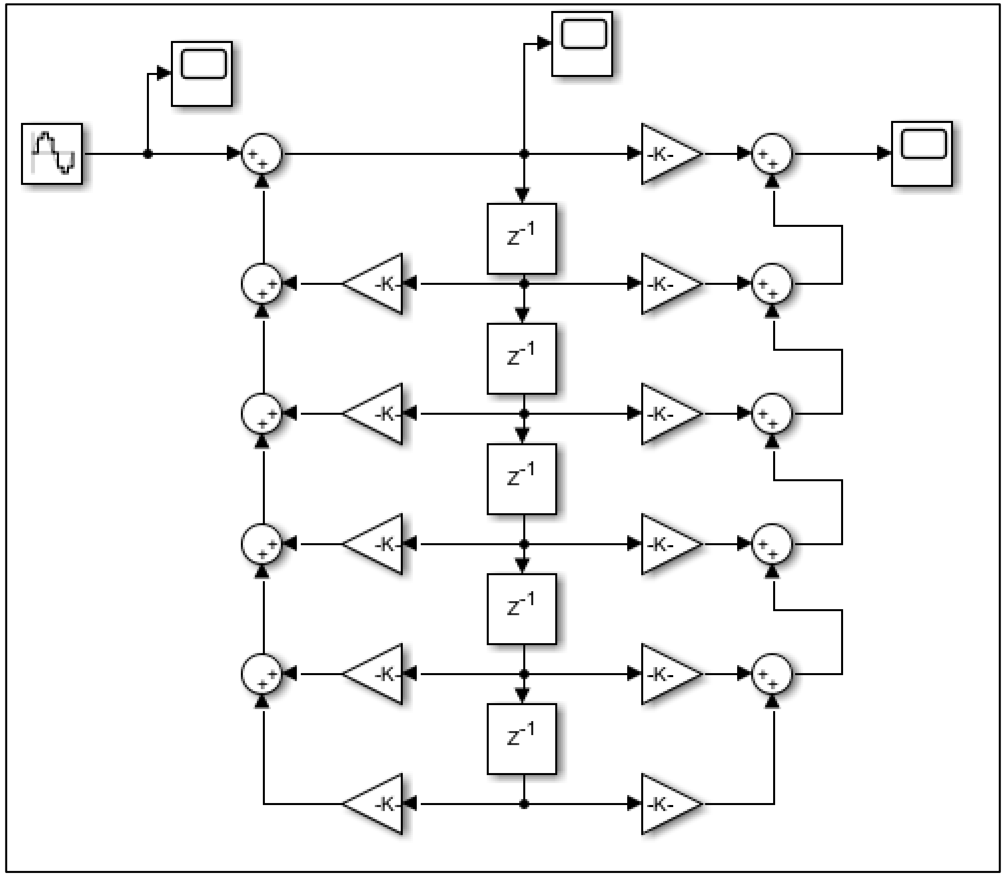
Set time (how long)



Click it to run the simulation



Direct Type I



Direct Type II

SDU

APP in Matlab- GUI