LECTURE 7: Damped sinusoids

7.1. RLC circuits

Input volatage: $u(t)$ ($u(t) = 0$ dla $t < 0$)

Output voltage: $u_C(t)$ (on the capacitor)

Fundamental relations:

$$i_c(t) = C \cdot du_C(t)/dt$$

$$u_L(t) = C \cdot di_L(t)/dt$$

From Kirchhoff law:

$$Ri(t) + L\frac{di(t)}{dt} + \frac{1}{C}\int i(t)\,dt = u_R(t) + u_L(t) + u_C(t) = u(t) \qquad (*)$$

Assuption:

zero initial voltage on the capacitor

Fourier transform of both sides of (*):

$$Ri(t) + L\frac{di(t)}{dt} + \frac{1}{C}\int i(t)\,dt = u(t)$$

$$\left[R + j\omega L + \frac{1}{j\omega C}\right] I(j\omega) = U(j\omega)$$

Only for the capacitance:

$$\frac{1}{C}\int i(t)dt = u_C(t)$$

$$\frac{1}{j\omega C} I(j\omega) = U_C(j\omega)$$

Inout / output relation:

$$H(j\omega)=\frac{U_C(j\omega)}{U(j\omega)}=\frac{\dfrac{1}{j\omega C}}{R+j\omega L+\dfrac{1}{j\omega C}}=\frac{1}{LC(j\omega)^2+RC(j\omega)+1}$$

$$H(j\omega)=\frac{1}{LC(j\omega)^2+RC(j\omega)+1}=\frac{\dfrac{1}{LC}}{(j\omega)^2+\dfrac{R}{L}(j\omega)+\dfrac{1}{LC}}$$

After new denotations:

$$\omega_0=1/\sqrt{LC},\quad \xi=(R/L)/(2\omega_0)$$

we get standard transfer function of the resonant system:

$$H(j\omega)=\frac{\omega_0^2}{(j\omega)^2+j2\xi\omega_0\omega+\omega_0^2}=\frac{1}{-(\omega/\omega_0)^2+j2\xi(\omega/\omega_0)+1}$$

Alternatively:

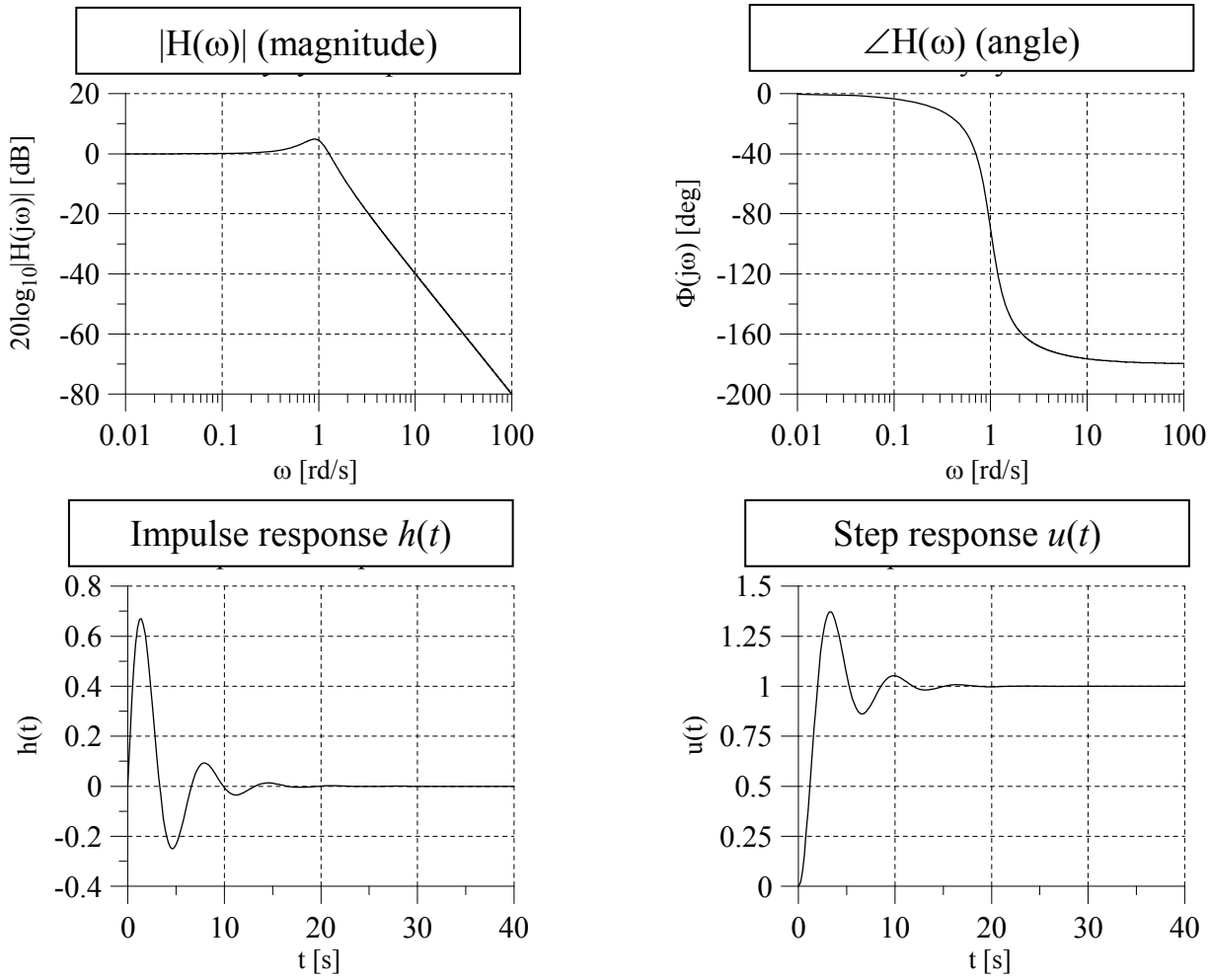
$$H(j\omega)=\frac{A\omega_1}{(a+j\omega)^2+\omega_1^2},\quad A=\frac{\omega_0}{\sqrt{1-\xi^2}},\quad \omega_1=\omega_0\sqrt{1-\xi^2},\quad a=\xi\omega_0$$

Impulse response:

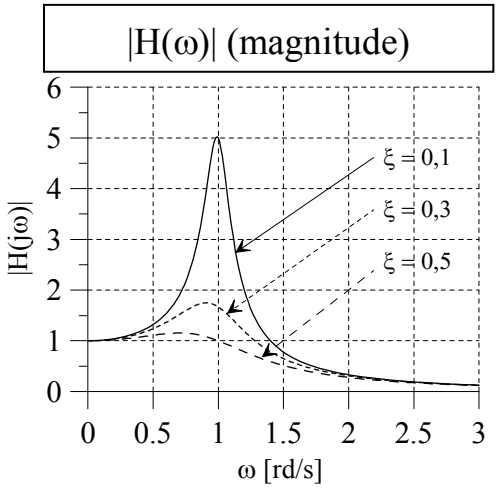
$$h(t)=\begin{cases}Ae^{-at}\sin(\omega_1 t) & \text{dla } t\ge 0\\ 0 & \text{dla } t<0\end{cases}$$

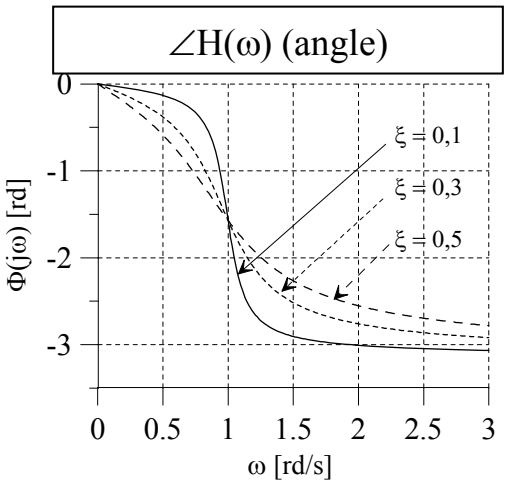
$$h(t)=\begin{cases}\dfrac{\omega_0}{\sqrt{1-\xi^2}}e^{-(\xi\omega_0)t}\sin\left(\omega_0\sqrt{1-\xi^2}\cdot t\right) & \text{dla } t\ge 0\\ 0 & \text{dla } t<0\end{cases}$$

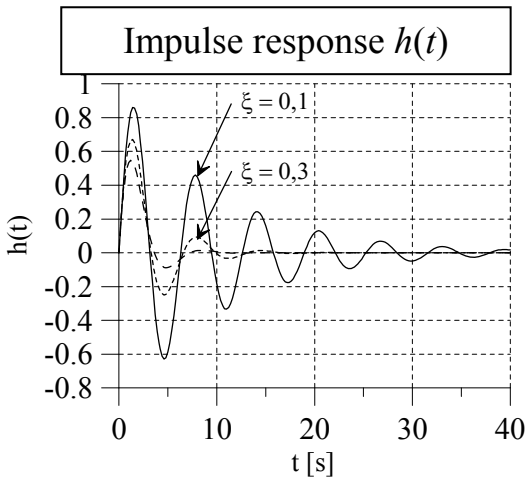
ω_1 – pusation of the RLC system = attenuated oscillations ($\omega_1 \neq \omega_0$ for $\xi\neq 0$)
ω_0 – eigen pusation of the un-attenuated system

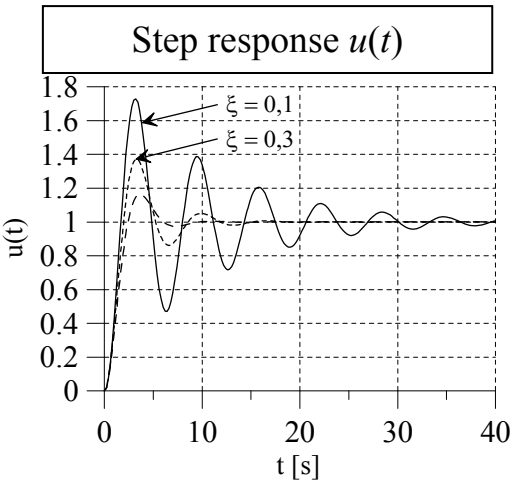


Maximum of $|H(\omega)|$ for $\omega = \omega_1$
Two poles = decreasing 2 * -20 dB = -40 dB per decade.


|H(ω)| (magnitude)
|H(jω)|
ω [rd/s]
ξ = 0,1
ξ = 0,3
ξ = 0,5


∠H(ω) (angle)
Φ(jω) [rd]
ω [rd/s]
ξ = 0,1
ξ = 0,3
ξ = 0,5


Impulse response h(t)
h(t)
t [s]
ξ = 0,1
ξ = 0,3


Step response u(t)
u(t)
t [s]
ξ = 0,1
ξ = 0,3

Fourier spectra:

1) Exponential signal :

$$x(t)=\begin{cases}0 & \text{dla } t<0\\ e^{-at} & \text{dla } t\geq 0\end{cases} \quad\leftrightarrow\quad H(j\omega)=\frac{1}{a+j\omega},\quad (a>0)$$

Proof: $\int_0^\infty e^{-at}e^{-j\omega t}dt=\int_0^\infty e^{-(a+j\omega)t}dt=\frac{1}{-(a+j\omega)}e^{-(a+j\omega)t}\Big|_0^\infty=\frac{1}{a+j\omega}$

2) Sinusoidal with exponential envelope:

$$x(t)=\begin{cases}0 & dla\ \ t<0\\ Ae^{-at}\sin\omega_0 t & dla\ \ t\geq 0\end{cases} \quad\leftrightarrow\quad X(j\omega)=\frac{A\omega_0}{(a+j\omega)^2+\omega_0^2},\quad (a>0)$$

Proof: $\int_0^\infty Ae^{-at}\sin(\omega_0 t)e^{-j\omega t}dt=A\int_0^\infty e^{-(a+j\omega)t}\frac{1}{2j}[e^{j\omega_0 t}-e^{-j\omega_0 t}]dt=$

$$=\frac{A}{2j}\left(\int_0^\infty e^{-[a+j(\omega-\omega_0)]t}dt-\int_0^\infty e^{-[a+j(\omega+\omega_0)]t}dt\right)=$$

$$=\frac{A}{2j}\left(\frac{1}{-(a+j(\omega-\omega_0))}e^{-(a+j(\omega-\omega_0))t}\Big|_0^\infty-\frac{1}{-(a+j(\omega+\omega_0))}e^{-(a+j(\omega+\omega_0))t}\Big|_0^\infty\right)=$$

$$=\frac{A}{2j}\left(\frac{1}{a+j(\omega-\omega_0)}-\frac{1}{a+j(\omega+\omega_0)}\right)=\frac{A}{2j}\frac{2j\omega_0}{(a+j(\omega-\omega_0))(a+j(\omega+\omega_0))}=\frac{A\omega_0}{(a+j\omega)^2+\omega_0^2}$$

3) Sinusoidal with exponential envelope:

$$x(t)=\begin{cases}0 & \text{dla } t<0\\ Ae^{-at}\cos\omega_0 t & \text{dla } t\geq 0\end{cases} \quad\leftrightarrow\quad X(j\omega)=A\frac{a+j\omega}{(a+j\omega)^2+\omega_0^2},\ (a>0)$$

Proof: as above, setting $\cos(\omega_0 t) = 1/2(\exp(j\omega_0 t)+\exp(-j\omega_0 t))$.

7.2. Hilbert transform method

Continuous Hilbert transform of the real signal $x_r(t)$

$$x_i(t) = H[x_r(t)] = \frac{1}{\pi}\int_{-\pi}^{\pi}\frac{x_r(\tau)}{t-\tau}d\tau \quad \leftrightarrow \quad X_i(j\omega) = H(j\omega)X_r(j\omega)$$

$x_r(t)$ *is convolved with* $h(t) = \frac{1}{\pi t}$: $$x_i(t) = \int_{-\infty}^{+\infty} x_r(\tau)h(t-\tau)d\tau$$

Inverse continuous Hilbert transform of the real signal $x_i(t)$

$$x_r(t) = H^{-1}[x_i(t)] = -\frac{1}{\pi}\int_{-\pi}^{\pi}\frac{x_i(\tau)}{t-\tau}d\tau \quad \leftrightarrow \quad X_r(j\omega) = H^{-1}(j\omega)X_i(j\omega)$$

$x_i(t)$ is convolved with $h(t) = -\frac{1}{\pi t}$: $$x_r(t) = \int_{-\infty}^{+\infty} x_i(\tau)h(t-\tau)d\tau$$

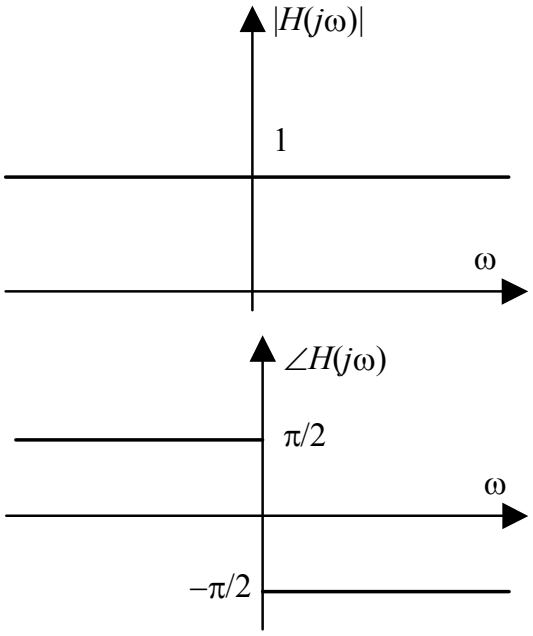
Transfer functions:

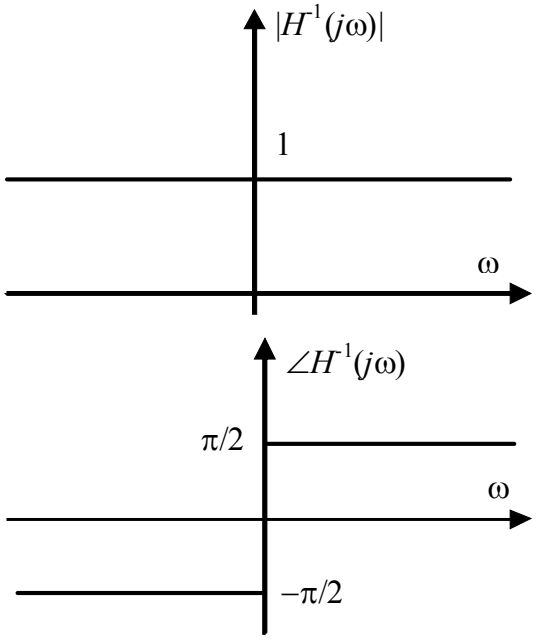
Hilbert transform
phase shifter $-\pi/2$

$$H(j\omega)=\begin{cases}-j, & \omega>0\\ 0, & \omega=0\\ j, & \omega<0\end{cases},$$

Inverse Hilbert transform
phase shifter $+\pi/2$

$$H^{-1}(j\omega)=\begin{cases}j, & \omega>0\\ 0, & \omega=0\\ -j, & \omega<0\end{cases}$$





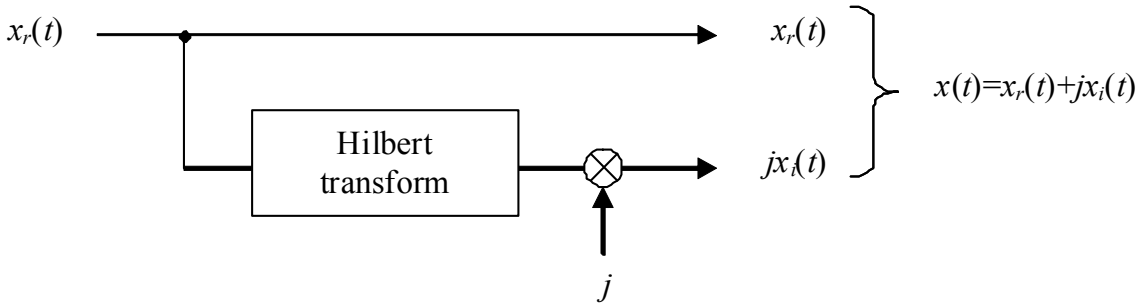
Hilbert transform: $x_r(t)=\cos(\omega_0 t) \Rightarrow x_i(t)=\sin(\omega_0 t)$

$$x_r(t)=\cos(\omega_0 t)$$

$$x_i(t)=\cos(\omega_0 t-\pi/2)=\cos(\omega_0 t)\cos(\pi/2)+\sin(\omega_0 t)\sin(\pi/2)=\sin(\omega_0 t)$$

$$x_r(t)=\sin(\omega_0 t+\pi/2)=\sin(\omega_0 t)\cos(\pi/2)+\cos(\omega_0 t)\sin(\pi/2)=\cos(\omega_0 t)$$

Analytic signal:



$$x(t)=x_r(t)+jx_i(t)=\cos(\omega_0 t)+j\sin(\omega_0 t)=e^{j\omega_0 t}$$

Derivation of Hilbert transfer function:

Since (sgn = sign):

$$x(t) = \mathrm{sgn}(t) \quad \leftrightarrow \quad X(j\omega) = \frac{2}{j\omega}$$

and Fourier transform has duality property:

$$X(jt) \quad \leftrightarrow \quad 2\pi x(-\omega)$$

than

$$y(t) = \frac{2}{jt} \quad \leftrightarrow \quad Y(j\omega) = 2\pi\,\mathrm{sgn}(-\omega) = -2\pi\,\mathrm{sgn}(\omega)$$

Therefore:

$$h(t) = \frac{1}{\pi t} \quad \leftrightarrow \quad H(j\omega) = -j\,\mathrm{sgn}(\omega)$$

Fourier spectrum of the analytic signal:

$$x(t) = x_r(t) + j \cdot x_i(t) = x_r(t) + j \cdot H(x_r(t))$$

$$X(j\omega) = X_r(j\omega) + jX_i(j\omega) = X_r(j\omega) + jH(j\omega)X_r(j\omega) = [1 + jH(j\omega)]X_r(j\omega)$$

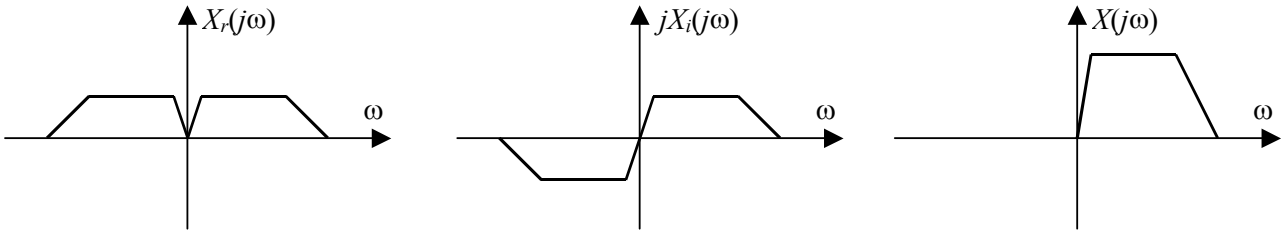
For $\omega \geq 0$:

$$X(j\omega) = [1 + j(-j)]X_r(j\omega) = 2X_r(j\omega)$$

For $\omega < 0$:

$$X(j\omega) = [1 + j(j)]X_r(j\omega) = [1-1]X_r(j\omega) = 0$$

Example:



Having analytic signal $x(t)$ we can reconstruct original signal $x_r(t)$ and its spectrum $X_r(j\omega)$:

Since:

$$\int_{-\infty}^{+\infty} x^*(t)e^{-j\omega t}dt = \left[\int_{-\infty}^{+\infty} x(t)e^{j\omega t}dt\right]^* = X^*(-j\omega)$$

therefore:

$$x_r(t) = \frac{1}{2}\left[x(t) + x^*(t)\right] \quad \leftrightarrow \quad X_r(j\omega) = \frac{1}{2}\left[X(j\omega) + X^*(-j\omega)\right]$$

$$x_i(t) = \frac{1}{2j}\left[x(t) - x^*(t)\right] \quad \leftrightarrow \quad X_i(j\omega) = \frac{1}{2j}\left[X(j\omega) - X^*(-j\omega)\right]$$

Discrete Hilbert Transform = discrete convolution

$$x_i(n) = \sum_{k=-\infty}^{\infty} x_r(k) h_H(n-k) = \sum_{k=-\infty}^{\infty} h_H(k) x_r(n-k)$$

Impulse response from frequency response:

$$h(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H\left(e^{j\Omega}\right) e^{j\Omega n} d\Omega, \quad -\infty \le n \le \infty$$

Impulse response of discrete Hilbert filter ($\Omega = 2\pi f/f_s$):

$$h_H(n) = \int_{-\pi}^{\pi} H_H(e^{j\Omega}) e^{j\Omega n} d\Omega$$

$$h_H(n) = \frac{1}{2\pi} \int_{-\pi}^{0} j e^{j\Omega n} d\Omega + \frac{1}{2\pi} \int_{0}^{\pi} (-j) e^{j\Omega n} d\Omega = \frac{j}{2\pi} \left[\frac{1}{jn} e^{j\Omega n} \Big|_{-\pi}^{0} - \frac{1}{jn} e^{j\Omega n} \Big|_{0}^{\pi} \right]$$

$$= \frac{1}{2\pi n} \left[(e^{j0} - e^{-j\pi n}) - (e^{j\pi n} - e^{j0}) \right] = \frac{1}{2\pi n} \left[2 - 2\cos \pi n \right] = \frac{1}{\pi n} \left[1 - \cos \pi n \right]$$

Since:

$$\sin^2 \alpha = \frac{1 - \cos 2\alpha}{2}$$

Then finally:

$$h_H(n) = \frac{2\sin^2(\pi n/2)}{\pi n} = \frac{\sin^2(\pi n/2)}{\pi n/2}$$

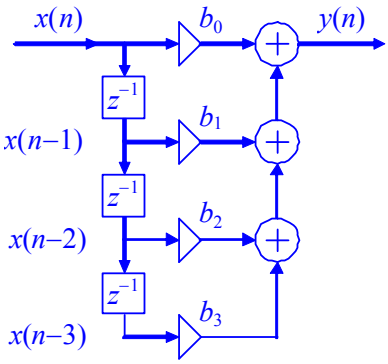
$$h_H(n) = \begin{cases} \dfrac{\sin^2(\pi n/2)}{\pi n/2}, & n \neq 0 \\ 0, & n = 0 \end{cases}$$

Multiplication with symmetrical window $w(n)$:

$$h_H^{(w)}(n) = h_H(n) w(n), \qquad n = -M, \ldots -1, 0, 1, \ldots, M$$

Delay M samples:

$$b_n = h_H^{(w)}(n - M), \qquad n = 0, 1, 2, \ldots, 2M$$

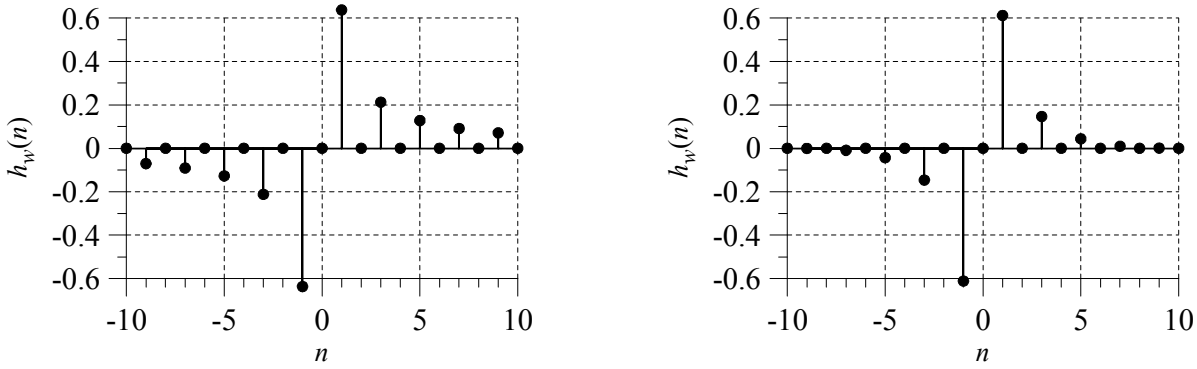


Digital Hilbert filter $h_H^{(w)}(n) = h_H(n)w(n)$

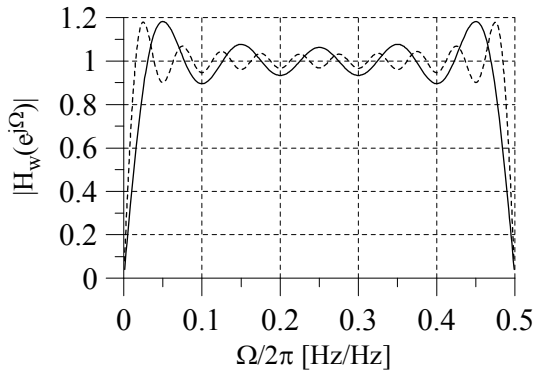
Solid line: $M = 10$ ($N = 2M+1 = = 21$).
Dashed line: $M = 20$ ($N = 2M+1 = 41$)
$\Omega = 2\pi f/f_s$

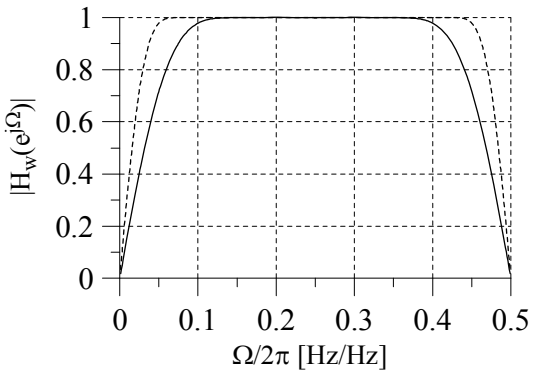
with rectangular window **with Blackman window**

impulse response $h_H^{(w)}(n)$

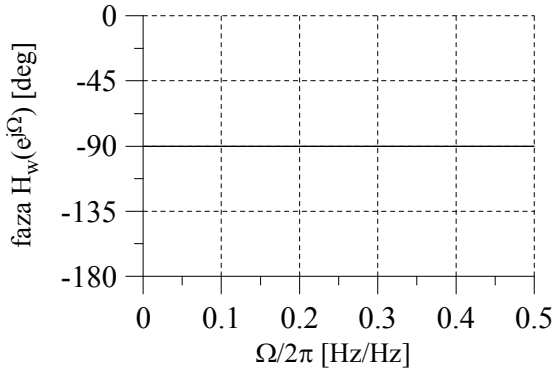


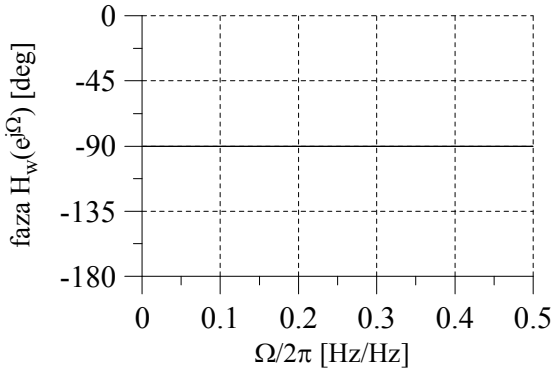
$|H_H^{(w)}(e^{j\Omega})|$ (magnitude)





$|H_H^{(w)}(e^{j\Omega})|$ (angle, phase)





Hilbert transform calculation

1) In time as a convolution:

$$x_i(n) = \sum_{k=0}^{2M} b_k x_r(n-k) = \sum_{k=0}^{2M} h_H^{(w)}(k-M) x_r(n-k)$$

2) In frequency via spectrum modification:

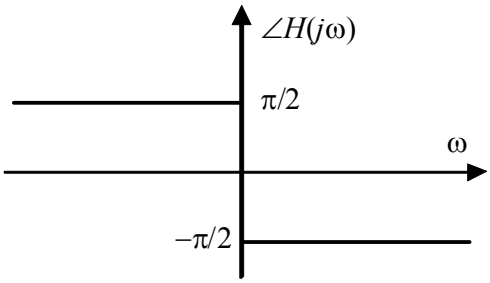
$$n = 0, 1, 2, ..., N-1, \qquad k = 0, 1, 2, ..., N-1, \qquad \Omega_k = k2\pi / N$$

$$X_r(k) = X_r(e^{j\Omega k})$$

Method 1 – imaginary complement

$$x_r(n) \xrightarrow{FFT(N)} X_r(e^{j\Omega_k}) \rightarrow X_i(e^{j\Omega_k}) = H(e^{j\Omega_k}) X_r(e^{j\Omega_k}) \xrightarrow{FFT^{-1}(N)} x_i(n)$$

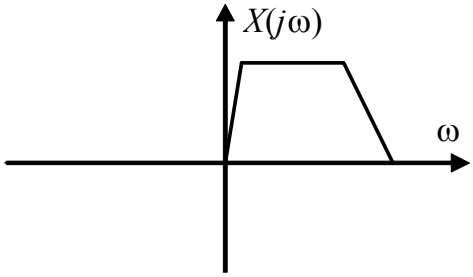
$$H(k) = H(e^{j\Omega_k}) = \begin{cases} 0, & k = 0 \\ -j, & k = 1...(N/2-1) \\ 0, & k = N/2 \\ j, & k = (N/2)...(N-1) \end{cases}$$



Method 2 – analytic signal (function `hilbert()` in Matlab)

$$x_r(n) \xrightarrow{FFT(N)} X_r(e^{j\Omega_k}) \rightarrow X(e^{j\Omega_k}) = W(e^{j\Omega_k}) X_r(e^{j\Omega_k}) \xrightarrow{FFT^{-1}(N)} x(n)$$

$$W(k) = W(e^{j\Omega_k}) = \begin{cases} 1, & k = 0 \\ 2, & k = 1...(N/2-1) \\ 1, & k = N/2 \\ 0, & k = (N/2)...(N-1) \end{cases}$$



Hilbert transform applications

1) Instantaneous amplitude, phase, frequency demodulation:

Input: $x_r(t) = A(t)\cos(\phi(t)), \quad A(t) > 0, \phi(t) -$ varying

After Hilbert: $x(t) = A(t)e^{j\phi(t)}$

Result: $A(t) = |x(t)|,$ e.g. $A(t) = e^{-at} \rightarrow a = ?$

$$\varphi(t) = \tan\frac{\mathrm{Im}\left(x(t)\right)}{\mathrm{Re}\left(x(t)\right)} \quad \text{e.g.} \quad \varphi(t)=\omega_0 t + \varphi_0$$

$$\omega(t) = \frac{d\varphi(t)}{dt} \qquad \text{e.g.} \quad \omega(t) = \frac{d\left[\omega_0 t + \varphi_0\right]}{dt} = \omega_0 = ?$$

2) Instantaneous phase difference demodulation:

Input:

$x_1(t) = A_1(t)\cos(\omega t + \phi_1(t))$, $A_1(t)>0$ slowly varying

$x_2(t) = A_2(t)\cos(\omega t + \phi_2(t))$, $A_1(t)>0$ slowly varying

After Hilbert:

$$y_1(t) = A_1(t)\,e^{j\left(\omega t+\phi_1(t)\right)},$$

$$y_2(t) = A_2(t)e^{j(\omega t+\phi_2(t))}$$

$$z(t) = y_1^*(t)y_2(t) = A_1(t)A_2(t)e^{j\left(-\omega t-\phi_1(t)+\omega t+\phi_2(t)\right)} = A_1(t)A_2(t)e^{j\left(\phi_2(t)-\phi_1(t)\right)}$$

Result: $$\Delta\phi(t) = \phi_2(t) - \phi_1(t) = \mathrm{arctg}\left(\frac{\mathrm{imag}(z(t))}{\mathrm{real}(z(t))}\right)$$

In case of noisy signals:

$$z(t) = \int_{-T}^{T} w(\tau)\,y_1^*(t+\tau)\,y_2(t+\tau)\,d\tau, \qquad \int_{-T}^{T} w(\tau)\,d\tau = 1$$

Matlab program 1: amplitude and frequency demodulation

```
N=1000; fpr=1000;
dt=1/fpr; t=dt*(0:N-1);
xr = (1+0.25*sin(2*pi*1*t)) .* cos(2*pi*(50*t+0.5*200*t.^2);
x = hilbert(xr);
ampl = abs(x);
faza = unwrap(angle(x));
freq = 1/(2*pi)*diff(faza)./diff(t);
plot(t,xr); pause
plot(t,ampl); pause
plot(t,faza); pause
plot(t(1:N-1),freq); pause

% ######################
% OUR Hilbert transform
% modification 2
% ######################

Xr = fft(xr);
PhaseMod = [ 0 -j*ones(1,Nx/2-1) 0 j*ones(1,Nx/2-1) ];
X = Xr .* PhaseMod;
xi = ifft(X);
xour = xr + j*xi;
plot(t,xr,'r',t,xi,'b'); title('MY: RED=real, BLUE=imag');
grid; pause
error = max(abs(x-xour)), pause % error in respect to Matab
```

Matlab program 2: parameters of damped sinusoid

```
% Parameters values
fs =2000;                   % sampling freq [Hz]
T = 2^4;                    % observation time [s]
dt = 1/fs;                  % sepling pperiod [s]
t = 0 : dt : L-dt;          % discretized time
N = length(t),              % number of samples
A0 = 4.5;                   % max (starting) amplitude
d = 0.0001;                 % logarithmic decrement
f0 = 32.77*(fs/N);          % resonant frequency [Hz]

w0 = 2*pi*f0;
a = d*f0;

% Signal generation
x = A0* exp(-a*t) .* cos(2*pi*f0*t);
% Add AWGN noise
x = awgn(x,38);             % SNR = 38 dB
```

```
y = hilbert(x);
ylog = log(abs(y));

% Cut a fragment from the central part of y
Nc = round(N/2+1);  M=round(N/16);
n = Nc - M : Nc + M;
y = y(n);
ylog = ylog(n);
tn = t(n);
subplot(111); stem(ylog); grid; pause

% Estimate frequency
phn = unwrap( atan2(imag(y),real(y)) );
fn = diff(phn)./diff(tn)/(2*pi);
f0_est = mean(fn)

% Estimate logarithmic decrement – attenuation speed
p = polyfit(tn,ylog,1);
d_est = -p(1)/f0_est

disp('#####################################################')
err_f0 = 100*abs(f0-f0_est)/f0
err_d = 100*abs(d-d_est)/d
disp('#####################################################')
```

7.3. Fourier transform methods

RLC circuit:

$$h(t)=\begin{cases}\dfrac{\omega_0}{\sqrt{1-\xi^2}}e^{-(\xi\omega_0)t}\sin\left(\omega_0\sqrt{1-\xi^2}\cdot t\right) & \text{dla } t\geq 0\\ 0 & \text{dla } t<0\end{cases}$$

Continuous signal model:

$$x(t)=Ae^{-\delta f_0 t}\cos(2\pi f_0 t+\varphi)+\varepsilon_w(t)+\varepsilon_{ZPD}(t)$$

$\varepsilon_w(t)$ = AWGN,… $\varepsilon_{ZPD}(t)$ – Zero Point Drift (DC drift)

Discrete signal model:

$$x[n]=Ae^{-\beta n}\cos(\omega_0 n+\varphi)+\varepsilon_w[n]+\varepsilon_{ZPD}[n]$$

$$\beta=\delta f_0/f_s,\qquad \omega_0=2\pi(f_0/f_s)$$

Optimization = error minimization:

$$C(A,\beta,\omega_0,\varphi,\varepsilon_{dc})=\sum_{n=0}^{N-1}[x[n]-Ae^{-\beta n}\cos(\omega_0 n+\varphi)-\varepsilon_{dc}]^2\xrightarrow[\beta,\,\omega_0=?]{}\min$$

Multiplication by a window:

$$v[n]=w[n]x[n]=w[n]Ae^{-\beta n}\cos(\omega_0 n+\varphi)\qquad (*)$$

N-point DFT (FFT) of $v[k]$:

$$V[k]=\sum_{n=0}^{N-1}v[n]e^{-j\omega_k n},\quad \omega_k=\frac{2\pi}{N}k,\quad k=0,1,2,\ldots,N-1$$

$$\omega_0=(k\pm d)\frac{2\pi}{N},\quad 0<d\leq 0.5$$

For $w[n]$ = rectangular window in (*):

$$V[k]=\frac{A}{2}\left(e^{j\varphi}\frac{1-\lambda^N}{1-\lambda e^{-j\omega_k}}+e^{-j\varphi}\frac{1-\lambda^{*N}}{1-\lambda^* e^{-j\omega_k}}\right)$$

where: $\lambda=e^{-\beta+j\omega_0}$, $\omega_k=(2\pi/N)k$, "*" - complex conjugattion.

We assume that the spectral leakage from the negative frequencies can be neglected. Therefore in **Duda-1 algorithm (Bertocco-Yoshida-1)**:

$$V[k]=\frac{A}{2}\left(e^{j\varphi}\frac{1-\lambda^N}{1-\lambda e^{-j\omega_k}}+e^{-j\varphi}\frac{1-\lambda^{*N}}{1-\lambda^* e^{-j\omega_k}}\right)$$

$$\Downarrow$$

$$V[k]\approx\frac{A}{2}\left(e^{j\varphi}\frac{1-\lambda^N}{1-\lambda e^{-j\omega_k}}\right)$$

$$\Downarrow \qquad (\max \text{ in } V[k])$$

$$R=\frac{V[k-1]-V[k]}{V[k]-V[k+1]}=\frac{1-\lambda e^{-j\omega_{k+1}}}{1-\lambda e^{-j\omega_{k-1}}}r, \qquad r=\frac{-e^{-j\omega_k}+e^{-j\omega_{k-1}}}{-e^{-j\omega_{k+1}}+e^{-j\omega_k}}$$

$$\Downarrow$$

$$\lambda=e^{j\omega_k}\frac{r-R}{re^{-j2\pi/N}-Re^{j2\pi/N}}$$

$$\Downarrow$$

$$\omega_0=\mathrm{Im}\{\ln(\lambda)\} \quad \text{and} \quad \beta=-\mathrm{Re}\{\ln(\lambda)\}$$

$$\Downarrow$$

$$f_0=\frac{\omega_0}{2\pi}f_s, \qquad \delta=\beta\frac{f_s}{f_0}$$

TABLE
THE SUMMARY OF THE DFT INTERPOLATION FORMULAS

Method	Ratio of DFT bins $V[k]$	Resonant frequency $f_0=f_s\,(k\pm d)/N$ or $f_0=f_s\,\omega_0/(2\pi)$	Logarithmic decrement $\delta=\beta f_s/f_0$
Bertocco (BY-0) **[1]**	rectangular window $R=\frac{V[k\pm1]}{V[k]}$, $z=\frac{1-R}{1-R\exp(\pm(-j2\pi/N))}$	$d=\frac{N}{2\pi}\arg\{z\}$	$\beta=\ln\lvert z\rvert$
Duda-1 (BY-1) **[2]**	rectangular window $R=\frac{V[k-1]-V[k]}{V[k]-V[k+1]}$ $\lambda=e^{j\omega_k}\frac{r-R}{re^{-j2\pi/N}-Re^{j2\pi/N}}$ r given by (13)	$\omega_0=\mathrm{Im}\{\ln(\lambda)\}$	$\beta=-\mathrm{Re}\{\ln(\lambda)\}$
Yoshida (BY-2) **[3]**	rectangular window $R=\frac{V[k-2]-2V[k-1]+V[k]}{V[k-1]-2V[k]+V[k+1]}$	$d=\mathrm{Re}\{3/(R-1)\}$	$\beta=\frac{2\pi}{N}\mathrm{Im}\{-3/(R-1)\}$
Duda-3 (BY-3) **[2]**	rectangular window $R=\frac{V[k-2]-3V[k-1]+3V[k]-V[k+1]}{V[k-1]-3V[k]+3V[k+1]-V[k+2]}$ $\lambda=e^{j\omega_k}\frac{r-R}{re^{-j2(2\pi/N)}-Re^{j2(2\pi/N)}}$ r given by (A7)	$\omega_0=\mathrm{Im}\{\ln(\lambda)\}$	$\beta=-\mathrm{Re}\{\ln(\lambda)\}$
Agrež **[4]**	Hann window (RVCI, M=1) $R=2\frac{\lvert V[k+1]\rvert-\lvert V[k-1]\rvert}{2\lvert V[k]\rvert+\lvert V[k-1]\rvert+\lvert V[k+1]\rvert}$	$d=R$ (three-point interpolation for undamped sinusoids)	Same as proposed RVCI-M for M=0 or M=1
Duda-*M* (RVCI-M) **[2]**	RVCI window, arbitrary order M $R_1=\frac{\lvert V[k+1]\rvert^2}{\lvert V[k]\rvert^2}$, $R_2=\frac{\lvert V[k-1]\rvert^2}{\lvert V[k]\rvert^2}$	$d=-\frac{2M+1}{2}$ $\frac{R_1-R_2}{2(M+1)R_1R_2-R_1-R_2-2M}$	$\beta=\frac{2\pi}{N}\sqrt{\frac{(d+M)^2-R_1(d-M-1)^2}{R_1-1}}$, $d\neq0.5$

$$w_M[n]=\begin{cases}\sum_{m=0}^{M}(-1)^m A_w[m]\cos\left(\frac{2\pi}{N}mn\right)\\ 0,\quad \text{otherwise}\end{cases}$$

RIFE-VINCENT CLASS I (RVCI) COEFFICIENTS

$m=$	0	1	2	3	4	5	6
A_m^w, M=0	1						
A_m^w, M=1	1	1					
A_m^w, M=2	1	4/3	1/3				
A_m^w, M=3	1	3/2	3/5	1/10			
A_m^w, M=4	1	8/5	4/5	8/35	1/35		
A_m^w, M=5	1	105/63	60/63	45/126	5/63	1/126	
A_m^w, M=6	1	396/231	495/462	110/231	33/231	6/231	1/462

[1] M. Bertocco, C. Offeli, D. Petri, "Analysis of damped sinusoidal signals via a frequency-domain interpolation algorithm," *IEEE Trans. Instrum. Meas.*, vol. 43, no. 2, pp. 245-250, 1994.

[2] I. Yoshida, T. Sugai, S. Tani, M. Motegi, K. Minamida H. Hayakawa, "Automation of internal friction measurement apparatus of inverted torsion pendulum type," *J. Phys. E: Sci. Instrum.*, vol. 14, pp. 1201-1206, 1981.

[3] K. Duda, M. Magalas, M. Majewski, T. Zieliński: „DFT-based Estimation of Damped Oscillation's Parameters in Low-frequency Mechanical Spectroscopy", IEEE Trans. on Instrumentation and Measurement, 2011, in print.

[4] D. Agrež, "A frequency domain procedure for estimation of the exponentially damped sinusoids," *International Instrumentation and Measurement Technology Conference*, May 2009.

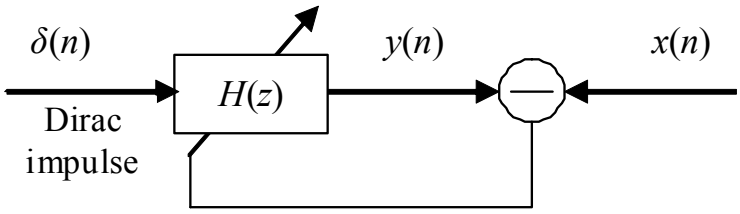
Matlab program 3: parameters of damped sinusoid via IpDFT

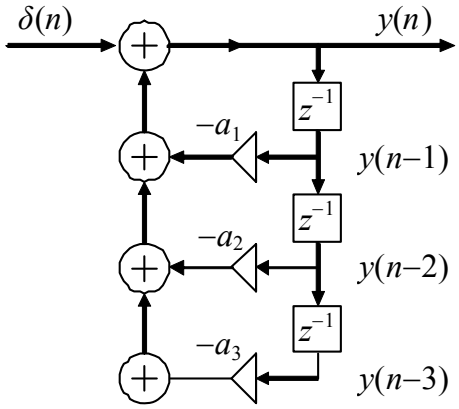
```
format long
% Parameters values …the same as in program2
% Signal generation ….the same as in program2
% Add AWGN noise…the same as in program2

% Yoshida 1981 - FFT with rectangular window
Y = fft(x,N)/N;                          % FFT
[Ymax s]=max(abs(Y));                    % find maximum
Y = Y/Ymax;                              %
if( abs(Y(s-1)) > abs(Y(s+1)) )  ss=s-1;
else ss=s;
end

% Real maximum should be between freq samples 2<=no<3
s1=ss-1;
s2=ss;
s3=s2+1;
s4=s3+1;
R=(Y(s1)-2*Y(s2)+Y(s3))/(Y(s2)-2*Y(s3)+Y(s4));
d_est1 = 2*pi*(imag(-3/(R-1))./real((s1-1)-3/(R-1)));
f_est1 =(fs/N)*real((s1-1)-3/(R-1);
err_d1  = 100*abs(d-d_est1  )/d;
err_f1  = 100*(f0 - f_est)/f;

% Agres 2009: FFT with Hanning window
w = hanning(N)';
Yw = fft(w.*x,NFFT)/NFFT;       % FFT(signal*window)
Yw = abs(Yw);                   %
[Ywmax sw]=max(Yw);             % find maximum
Yw = Yw/Ywmax;                  % scale
% Yw can be calculated from Y (Yoshida metod) by local smoothing [1/4, 1/2, 1/4]
s1w = sw-1,
s2w = sw,
s3w = sw+1,
Y1w = Yw(s1w),
Y2w = Yw(s2w),
Y3w = Yw(s3w),
R = 2*( Y3w - Y1w ) / ( Y1w + 2*Y2w + Y3w ), pause
d_est2 = R;
k_est2 = (sw-1) + delta;
f_est2 = f0*k_est2;
err_d2 = 100*(d - d_est2)/d;
err_f2 = 100*(f0 - f_est2)/f;
```

7.4. LS and Prony methods

Design of AR (IIR) digital FILTER

- its impulse response is equal to $x(n)$
- its frequency response fits to signal spectrum
- its parameters → parameters of damped sinusoids





$$y(n) = \delta(n) - \sum_{k=1}^{N} a_k y(n-k)$$

When $y(n) \approx x(n) \quad \Rightarrow \quad x(n) = \delta(n) - \sum_{k=1}^{N} a_k x(n-k)$ linear self-prediction

Transfer function of digital filter:

$$H(z) = \frac{1}{1+\sum_{k=1}^{N} a_k z^{-k}} = \frac{1}{1+a_1 z^{-1}+a_2 z^{-2}+\ldots+a_N z^{-N}} = \frac{1}{(1-p_1 z^{-1})(1-p_2 z^{-1})\ldots(1-p_N z^{-1})}$$

For $z = e^{j\Omega} = e^{j2\pi\frac{f}{f_s}} \quad H(z) \to H(f).$

Task: find poles of the transfer function for a given signal *x(n)*

$$p_k = r_k \cdot e^{j\varphi_k} = r_k \cdot e^{j2\pi\frac{f_k}{f_s}} \quad \Rightarrow \quad h_k(n) = (p_k)^n$$

$$x(n) = \sum_{k=1}^{N/2}\left[(p_k)^n + (p_k^*)^n\right] = \sum_{k=1}^{N/2}\left[r_k^n \cdot e^{j\varphi_k n} + r_k^n \cdot e^{-j\varphi_k n}\right] = \sum_{k=1}^{N/2}\left[2r_k^n \cdot \cos\left(2\pi\frac{f_k}{f_s}n\right)\right]$$

$$x(n) = \sum_{k=1}^{N/2}\left[2e^{\ln(r_k)n} \cdot \cos(\varphi_k n)\right], \quad a_k = -\ln(r_k), \quad f_k = \frac{\varphi_k}{2\pi} f_s$$

Linear self-prediction:

$$\begin{bmatrix} a_p & a_{p-1} & \cdots & a_1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & a_p & a_{p-1} & \cdots & a_1 & 1 & 0 & \cdots & 0 \\ \vdots & & & & & & & & \\ 0 & 0 & \cdots & 0 & a_p & a_{p-1} & \cdots & a_1 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{bmatrix} = \mathbf{0}$$

$\mathbf{Ax} = \mathbf{0}$, **A** – matrix with Toeplitz structure

$$\begin{bmatrix} x_0 & x_1 & \cdots & x_p \\ x_1 & x_2 & \cdots & x_{p+1} \\ \vdots & & & \\ x_{N-1-p} & x_{N-2-p} & \cdots & x_{N-1} \end{bmatrix} \begin{bmatrix} a_p \\ a_{p-1} \\ \vdots \\ a_1 \\ 1 \end{bmatrix} = 0$$

$\mathbf{Xa} = \mathbf{0}$, **X** – matrix with Hankel structure

$$\begin{bmatrix} x_0 & x_1 & \cdots & x_{p-1} \\ x_1 & x_2 & \cdots & x_{p-2} \\ \vdots & & & \\ x_{N-1-p} & x_{N-2-p} & \cdots & x_{N-2} \end{bmatrix} \begin{bmatrix} a_p \\ a_{p-1} \\ \vdots \\ a_1 \end{bmatrix} = \begin{bmatrix} -x_p \\ -x_{p-1} \\ \vdots \\ -x_{N-1} \end{bmatrix}$$

$\mathbf{Xa} = \mathbf{x}$, **X** – matrix with Hankel structure

ALGORITH:

1) Solve the above equation in respect to **a**.
2) Find roots of p_k polynomial having coeffs **a.**
3) Knowing p_k calculate parameters of damped sins $e^{-a_k n}\cos\left(2\pi\frac{f_k}{f_s}n\right)$

$$p_k = r_k \cdot e^{j\varphi_k} = r_k \cdot e^{j2\pi\frac{f_k}{f_s}} \quad \Rightarrow \quad a_k = -\ln(r_k), \quad f_k = \frac{\varphi_k}{2\pi} f_s$$

$$x(n) = \sum_{k=1}^{N/2}\left[2e^{\ln(r_k)n}\cdot\cos\left(\varphi_k n\right)\right], \quad a_k = -\ln(r_k), \quad f_k = \frac{\varphi_k}{2\pi} f_s$$

```
% LS solution of the AR filter of the second order (x is vertical)
xm0   = x(3:end-0);
xm1 = x(2:end-1);
xm2 = x(1:end-2);
a = [ -xm1(:) –xm2(:) ] \ xm0(:);   % find LS solution
r = roots([1 a(1) a(2)]);
a_LS_est = abs(r(1));
d_LS_est = -log( a_est );
d_LS_est = d_LS_est*(length(x)-1)/f0/t(length(x)); % scaling
f_LS_est = fs*angle(p)/(2*pi);

% Prony solution – find AR filter of the second order corresponding to x(n)
[b,a] = prony(x,1,2);            % printsys(b,a,'z')
[z,p,K] = TF2ZPK(b,a);           % {a,b} → {zeros, poles}
a_est = abs(p(1));
d_est = -log( a_est );
d_est = d_est*(length(x)-1)/f0/t(length(x));    % scaling
f_est = fs*angle(p)/(2*pi);   % CHECK CORRECTNESS!
```

MATLAB function **[b,a] = prony(h, nb ,na)**

```
function [b,a] = prony(h, nb ,na)
% PRONY Prony's method for time-domain IIR filter design.
%  [B,A] = PRONY(H, NB, NA) finds a filter
%  with numerator order NB,
%  with denominator order NA,
%  and having the impulse response in vector h (real or complex).
%  The IIR filter coefficients are returned in
%  length NB+1 and NA+1 row vectors b and a,
%  ordered in descending powers of Z.
%
%   If the largest order specified is greater than
%   the length of H, H is padded with zeros.
%
%   References:
%     [1] T.W. Parks and C.S. Burrus, Digital Filter Design,
%         John Wiley and Sons, 1987, p226.

K = length(h) - 1;
M = nb; N = na;
if K <= max(M,N)      % zero-pad input if necessary
    K = max(M,N)+1;
    h(K+1) = 0;
end
c = h(1);
if c==0               % avoid divide by zero
    c=1;
end
H = toeplitz(h/c,[1 zeros(1,K)]);   % large Toeplitz matrix
% K+1 by N+1
if (K > N)
    H(:,(N+2):(K+1)) = [];
end
% Partition H matrix
H1 = H(1:(M+1),:);  % M+1 by N+1
h1 = H((M+2):(K+1),1);  % K-M by 1
H2 = H((M+2):(K+1),2:(N+1)); % K-M by N – cut Hankel matrix
a = [1; -H2\h1].';
b = c*a*H1.';
```

7.5. LP-SVD method - theory

R. Kumaresan, D. W. Tufts:
"Estimating the parameters of exponentially damped sinusoids and pole-zero modeling in noise"
IEEE Trans. Acoust. Speech Signal Processing, vol. ASSP-30, 837-840, 1982.

$$y(n) = \sum_{k=1}^{M} a_k \, e^{s_k n} + w(n) \qquad n = 0, 1, \cdots, N-1$$

$$\begin{bmatrix} y^*(1) & y^*(2) \cdots y^*(L) \\ y^*(2) & y^*(3) \cdots y^*(L+1) \\ \vdots & \vdots \quad \vdots \\ y^*(N-L) & y^*(N-L+1) \cdots y^*(N-1) \end{bmatrix} \begin{bmatrix} b(1) \\ b(2) \\ \vdots \\ b(L) \end{bmatrix} = - \begin{bmatrix} y^*(0) \\ y^*(1) \\ \vdots \\ y^*(N-L-1) \end{bmatrix}$$

$$B(z) = 1 + b(1)\, z^{-1} + b(2)\, z^{-2} + \cdots + b(L)\, z^{-L}$$

$$b = - \sum_{k=1}^{M} \sigma_k^{-1} \left[u_k^{\dagger} h \right] v_k$$

σ_k , k=1,2,…,L or N-L – singular values of $\mathbf{A}$
v_k, k=1,2,…,L – eigenvectors of $\mathbf{A}^H\mathbf{A}$
u_k, k=1,2,…,N-L – eigenvectors of $\mathbf{A}\mathbf{A}^H$
$\mathbf{A}^H\mathbf{A}$ – conjugation + transposition

LP-SVD method – MATLAB program

```
function [para] = lpsvd(y,M)
% LPSVD linear prediction with singular value decomposition
% function [para] = lpsvd(y,M)
% author: Yung-Ya Lin, 12/11/97
% reference: R. Kumaresan, D. W. Tufts IEEE Trans. Acoust. Speech Signal Processing
% vol. ASSP-30, 837-840, 1982.
% arguments:
% y: complex vector, NMR FID time series
% M: real scalar, number of signals or effective matrix rank
% para: real M*4 matrix, estimated damping factor, frequency, amplitude, phase

y=y(:);
N=length(y);                                          % # of complex data points in
FID
L=floor(N*3/4);                                       % linear prediction order L =
3/4*N
A=hankel(conj(y(2:N-L+1)),conj(y(N-L+1:N)));  % backward prediction data matrix
h=conj(y(1:N-L));                                     % backward prediction data vector
[U,S,V]=svd(A,0);                                     % singular value decomposition
clear A;
S=diag(S);
bias=mean(S(M+1:min([N-L,L])));                       % bias compensation
b=-V(:,1:M)*(diag(1./(S(1:M)-bias))*(U(:,1:M)'*h));   % prediction polynomial coeffs
s=conj(log(roots([b(length(b):-1:1);1])));            % polynomial rooting
s=s(find(real(s)<0));                                 % extract true signal poles
Z=zeros(N,length(s));
for k=1:length(s)
  Z(:,k)=exp(s(k)).^[0:N-1].';
end;
a=Z\y;                                                % linear least squares analysis
para=[-real(s) imag(s)/2/pi abs(a) imag(log(a./abs(a)))];
return
```