```
clc;
clear all;
close all;
format short;


%i.   Create input signal & parameters

lengthX = 5000;
res     = 1/5000;
T       = 1/400;
K       = 4;
minsep  = 100;
leftspace = lengthX - K*minsep;
randspace = rand(1,K);
randspace = floor(randspace * leftspace / sum(randspace));
nkOrig(1) = randspace(1);
for k = 2:K
%   nkOrig(k) = minsep*[1:K] + randspace;
    nkOrig(k) = nkOrig(k-1) + minsep + randspace(k);
end
%   nkOrig = [501 801 1201];
tkOrig     = (nkOrig-1) * res;
% akOrig   = ones(1,K);
akOrig     = normrnd(1,.2,1,K);
x          = zeros(1, lengthX);
x(nkOrig)  = akOrig;

tau        = lengthX*res;

% N = 32;
N = tau/T;
B = (N-(1-mod(N,2)))/tau;
M = floor(B*tau/2); % M = K;
L = floor((K+M)/2);


%Options
addNoise = 1;
SNR = -5;
useCadzow = 1;
cadzowIt = 5;
annih = 1; % annihilating filter or matrix pencil
numIt = 1;



%iii. Sampling
```

```
y = zeros(1,N);
y2 = zeros(1,N);
for k = 1:N
    y(k) = sum(akOrig.*diric(2*pi*(k*T-tkOrig), B));
    y2(k) = sum(akOrig.*sinc(B*(k*T-tkOrig)));
end



%iv.  Add noise

if addNoise
    sigma = sqrt(sum(y.^2)/(N*10^(SNR/10)));
    noise = sigma * randn(1,N);
    y = y + noise;
    y2 = y2 + noise;
end


%  Denoising using Cadzow


if useCadzow
    [X, Y] = meshgrid(exp(-j*2*pi/N*[1:N]), -M:1:M);
    dftMatrix = X .^ Y;
    yDFT = (dftMatrix * y.').';
    %(-M) -> 1
    %  k  -> k + M+1
    % (M) -> 2(M)+1
    yDFT = denoise_cadzow_sinc(yDFT, M, L, cadzowIt, K);
end


% Parameter retrieval using Annihilating filter


if annih
    h = annihilating_filter_asym(yDFT, K, 2*M-1);
    uk = roots(h);
else
    L = K+1; M = K;
%     M = round(length(yDFT)/2);
    L = length(yDFT)-M+1;
    uk = acmp(yDFT, K, L, M);
end

tk = real(log(uk)*tau/(-j*2*pi)).';
tk(tk<0) = tk(tk<0)+tau;
nk = round(tk / res) + 1;
```

```
ak = real( weights_asym(yDFT, exp(-j*2*pi*tk/tau), -M, 2*M+1) )*(T*B);

xRec = zeros(size(x));
xRec(nk) = ak;

% Display results

for (k = 1:length(tk))
    [val, index] = min((repmat(tk(k), size(tkOrig)) - tkOrig).^2);
    tkSort(index) = tk(k);
    akSort(index) = ak(k);
end

%Maximum likelihood estimation

t=mle(y);

if numIt == 1

    disp('%%--Stream of deltas--%%');
    disp(' ');
    disp(['-> Original tk   : ', mat2str(tkOrig)]);
    disp(['-> Estimated tk  : ', mat2str(tkSort)]);
    disp(['-> Squared Error : ', mat2str((tkOrig - tkSort).^2)]);
    disp(' ');

    disp(['-> Original ak   : ', mat2str(akOrig)]);
    disp(['-> Estimated ak  : ', mat2str(akSort)]);
    disp(['-> Squared Error : ', mat2str((akOrig - akSort).^2)]);


    figure, stem(x, '.', 'LineWidth', 2);
    figure, stem(t, '.', 'LineWidth', 2);
    ylim([min(0, 1.2*min(x)) max(0, 1.2*max(x))]);

    hold on, stem(xRec, '.--r')
    ylim([min(0, 1.2*min(xRec)) max(0, 1.2*max(xRec))]);

    xlim([0 lengthX]);

end
```