Appendix H: Program for designing with the aid of linear programming a filter considered in the lecture notes as well as in the handbook chapter

```
*****************************************************************************************
% This Matlab script-file optimizes the coefficients of
% of Example 4.11 in the handbook chapter T. Saramäki,
% "Finite Impulse Response Filter Design".
%
% Can be found in SUNs: ~ts/matlab/dsp/linear2.m
%
% Subfunctions needed: type1fre() (home-made)
%                      LP() (optimization toolbox)
%
% Copyright 1998 by Tapio Saramäki,
% Tampere University of Technology.

clear all
close all
NN=46 % Filter order
%--------------------------------------------------------------
% Half the filter order+1 (number of unknown filter coefficients)
%---------------------------------------------------------------
N=NN/2+1;
%--------------------------------
% Passband edge as fraction of pi
%--------------------------------
fp=.5;
%--------------------------------
% Stopband edge as fraction of pi
%--------------------------------
fs=.6;
%----------------------------------------------------
% Grid points: gridp and grids are the number of
% grid points in the passband and stopband, respectively
%----------------------------------------------------
gridp=20*N;
grids=20*N;
%----------------------------------------------------
% Desired response in the passband and stopband
%----------------------------------------------------
Des_pass=ones(1,gridp);
Des_stop=zeros(1,grids);
%----------------------------------------------------
% Passband and stopband weights
%----------------------------------------------------
```

```
Weig_pass=ones(1,gridp);
Weig_stop=sqrt(10)*ones(1,grids);
%******************************************************
% Set up matrises for Linear Programming:
% min z=c'x subject to Ax<=b
% Note that a linear combination of unknowns is
% minimized. In the lecture notes this combination
% was maximized!!
%******************************************************
% For our problem, x(i)=h(NN-i+1) for i=1,2,...,N, where
% N=NN/2+1 with NN being the filter order. x(N+1) is delta_s,
% the maximum absolute value of the weighted error function
% E(omega)=W(omega)[H(omega)-D(omega)], where H(omega)=
% x(1)*Phi(1,Omega)+...+x(N)*Phi(N,omega) with
% Phi(1,Omega)=1 and Phi(i,Omega)= 2*cos((i-1)*omega)
% for i > 1.
%******************************************************
wp=fp*pi;
ws=fs*pi;
%------------------------
% Passband grid points
% -----------------------
W=[0:wp/(gridp-1):wp];
%------------------------------
% Add the stopband grid points
%-------------------------------
W=[W ws:(pi-ws)/(grids-1):pi];
%***********************************************************************
%Frequency-domain conditions
%************************************************************************
% H(omega(i))-delta_s/W(omega(i)) <= D((omega(i))
% and
% -H(omega(i))-delta_s/W(omega(i)) <= -D((omega(i))
%*************************************************************************
for i=1:N
        G(:,i)=type1fre(i,W)';
end
A1=-1./[Weig_pass Weig_stop]';
A=[ G  A1
  -G  A1];
b=[Des_pass Des_stop -Des_pass -Des_stop]';
%*******************************************
% To be minimized is x(N+1)
%*******************************************
c=[zeros(1,N) 1];
%**************************************************************************
```

```
% Time-domain conditions: -0.05 <- x(N) <- +0.05,  -0.05 <- x(N)+x(N-1)
% <- +0.05, ..., -0.05 <- x(N)+x(N-1)+...+x(N-21) <- +0.05
%***************************************************************************
for k=1:22
clear A1
clear b1
clear A2
clear b2
b1=[0.05]';
b2=[0.05]';
A1=[zeros(1,N-k) ones(1,1+k-1) 0];
A2=[zeros(1,N-k) -ones(1,1+k-1) 0];
b=[b;b1;b2];
A=[A;A1;A2];
end
%***************************************************************************
% Condition 2*(x(N)+x(N-1)+...+x(2))+1=1
%***************************************************************************
clear A1
clear b1
clear A2
clear b2
b1=[1]';
b2=[-1]';
A1=[ones(1,1) 2*ones(1,N-1) 0];
A2=[-ones(1,1) -2*ones(1,N-1) 0];
b=[b;b1;b2];
A=[A;A1;A2];
%**************************************************
% Call LP function for Linear Programming
% This function is included in Optimization Toolbox
%**************************************************
x=lp(c,A,b);
%------------------------------------------
% h(NN-i+1)=x(i) for i=1,2,...N (N=NN/2+1)
% x(N+1) is the stopband ripple.
%------------------------------------------
%**************************************************
% Form the impulse response of the filter
%**************************************************
B=x(1:N)';
h=[fliplr(x(2:N)') B];
ripple=x(N+1)
[H,z]=zeroam(h,.0,1.,5000);
M=2*(N-1);
figure(1)
```

```
plot(z/pi,20*log10(abs(H)));axis([0 1 -80 10]);grid
title(['Amplitude response'])
xlabel('Angular frequency omega/pi');
ylabel('Amplitude in dB');
[H,z]=zeroam(h,.0,fp,2000);
figure(2)
dmin=1-min(H);dmax=max(H)-1;
plot(z/pi,H);axis([0 fp 1-1.05*dmin 1+1.05*dmax]);grid
title('Passband details')
xlabel('Angular frequency omega/pi');
ylabel('Amplitude');
hh=0
for k=1:47
hh=hh+h(k);
g(k)=hh;
end
for k=48:61
g(k)=hh;
end
figure(3)
impz(g);ylabel('Step response');xlabel('n in samples')
axis([0 60 -.1 1.1]);
```