Appendix A: Design of lowpass, highpass, bandpass, and bandstop filters using adjustable windows

```
********************************************************************************************
%-------------------------------------------------------
% Matlab m-file (firwinad.m)
% for designing a lowpass, highpass, bandpass, or
% bandstop linear-phase  FIR filter using the Kaiser,
% Saramaki, Dolph-Chebyshev, or transitional window.
% For highpass and bandstop filters, the order is
% restricted to be even.
%
% The filters are designed in such a way that the
% desired stopband or passband criterion just met
% and the other one is exceeded.
%
% Note that in the program N is the filter length
% so that N-1 is the order
%
% Tapio Saramäki  27.10.97
%
% This file can be found in SUN's:
% ~ts/matlab/dsp/firwinad.m
%-------------------------------------------------------
clear all;close all
disp('I am a program for designing a lowpass, highpass,')
disp('bandpass, or bandstop linear-phase FIR filter')
disp('with the aid of the Kaiser, Saramaki,')
disp('Dolph-Chebyshev, or transitional window')
disp('As input data, I first need the filter type')
disp('1 for lowpass, 2 for highpass');
firtyp=input('3 for bandpass, 4 for bandstop:');
%-------------------------------------------------------
if firtyp < 3
  wp=input('Passband edge as a fraction of pi: ');
  ws=input('Stopband edge as a fraction of pi: ');
  wpp=wp;wss=ws;
  if wp > ws;
    wpp=ws;
    wss=wp;
  end
end
%-------------------------------------------------------
if firtyp > 2
  wp(1)=input('First passband edge as a fraction of pi: ');
```

```
  wp(2)=input('Second passband edge as a fraction of pi: ');
  ws(1)=input('First stopband edge as a fraction of pi: ');
  ws(2)=input('Second stopband edge as a fraction of pi: ');
  wpp=wp(1);wss=ws(1);
  if wp(1) > ws(1);
    wpp=ws(1);
    wss=wp(1);
  end
  if abs(ws(2)-wp(2)) < abs(ws(1)-wp(1))
    wpp=wp(2);
    wss=ws(2);
    if wp(2) > ws(2);
      wpp=ws(2);
      wss=wp(2);
    end
  end
end
%------------------------------------------------------------------
dp=input('Passband ripple for the amplitude: ');
ds=input('Stopband ripple for the amplitude: ');
disp('Type 1 for Kaiser, 2 for Saramaki, 3 for')
itype=input('for Dolph-Chebyshev, or 4 for transitional window: ');
%------------------------------------------------------------------
% Estimate the length N; we start with odd length since highpass
% and bandpass filters must ne of odd length (odd order).
%------------------------------------------------------------------
if itype==1
  [N,alpha,wc]=kaiord(wpp,wss,dp,ds);
  N=2*floor((N-1)/2)-1;
end
if itype==2
  [N,beta,wc]=sarord(wpp,wss,dp,ds);
  N=2*floor((N-1)/2)-1;
end
if itype==3
  [N,beta,wc]=dchord(wpp,wss,dp,ds);
  N=2*floor((N-1)/2)-1;
end
if itype==4
  [N,beta,rho,wc]=traord(wpp,wss,dp,ds);
  N=2*floor((N-1)/2)-1;
end
%------------------------------------------------------------
% Find the filter coefficients h and the window coefficients
% wind for the estimated length N such that either the passband
% or stopband ripple criterion is just met and another one is
```

```
% less than the specified value. For a too low value of N,
% this is not true and N is increased. N is the length.
%------------------------------------------------------------
[h,wind]=winsubad(firtyp,itype,N,wp,ws,dp,ds);
%------------------------------------------------------
% Test whether the criteria at are met at the edges.
%------------------------------------------------------
isu=0;
if firtyp < 3
  [a,z]=zeroam(h,ws,ws,1);
  if abs(a) > ds isu=1;end
  [a,z]=zeroam(h,wp,wp,1);
  if abs(1-a) > dp isu=1;end
end
if firtyp > 2
  [a1,z]=zeroam(h,ws(1),ws(1),1);
  [a2,z]=zeroam(h,ws(2),ws(2),1);
  if max(abs(a1),abs(a2)) > ds isu=1;end
  [a1,z]=zeroam(h,wp(1),wp(1),1);
  [a2,z]=zeroam(h,wp(2),wp(2),1);
  if max(abs(1-a1),abs(1-a2)) > dp isu=1;end
end
increase=1;
if isu==0 increase=0;end
if isu==0 hs=h;winds=wind;NS=N;end
%------------------------------------------------------
% Increase the order by 2 until the criteria are met.
%------------------------------------------------------
if increase==1
  ll=0;
    while ll < 1
    N=N+2;
    [h,wind]=winsubad(firtyp,itype,N,wp,ws,dp,ds);
%------------------------------------------------------
% Test whether the criteria at are met at the edges.
%------------------------------------------------------
    isu=0;
if firtyp < 3
  [a,z]=zeroam(h,ws,ws,1);
  if abs(a) > ds isu=1;end
  [a,z]=zeroam(h,wp,wp,1);
  if abs(1-a) > dp isu=1;end
end
if firtyp > 2
  [a1,z]=zeroam(h,ws(1),ws(1),1);
  [a2,z]=zeroam(h,ws(2),ws(2),1);
```

```
  if max(abs(a1),abs(a2)) > ds isu=1;end
  [a1,z]=zeroam(h,wp(1),wp(1),1);
  [a2,z]=zeroam(h,wp(2),wp(2),1);
  if max(abs(1-a1),abs(1-a2)) > dp isu=1;end
end
    if isu==0 ll=1;
    hs=h;winds=wind;NS=N;end
    end
end
%-------------------------------------------------------
% Decrease the order by 2 until the criteria are just met
%-------------------------------------------------------
if increase==0
  ll=0;
    while ll < 1
    N=N-2;
    [h,wind]=winsubad(firtyp,itype,N,wp,ws,dp,ds);
%-------------------------------------------------------
% Test whether the criteria at are met at the edges.
%-------------------------------------------------------
    isu=0;
if firtyp < 3
  [a,z]=zeroam(h,ws,ws,1);
  if abs(a) > ds isu=1;end
  [a,z]=zeroam(h,wp,wp,1);
  if abs(1-a) > dp isu=1;end
end
if firtyp > 2
  [a1,z]=zeroam(h,ws(1),ws(1),1);
  [a2,z]=zeroam(h,ws(2),ws(2),1);
  if max(abs(a1),abs(a2)) > ds isu=1;end
  [a1,z]=zeroam(h,wp(1),wp(1),1);
  [a2,z]=zeroam(h,wp(2),wp(2),1);
  if max(abs(1-a1),abs(1-a2)) > dp isu=1;end
end
    ll=1;
    if isu==0 ll=0;
    hs=h;winds=wind;NS=N;end
    end
end
%-------------------------------------------------------
% Test whether the criteria are met by N=NS-1; only for
% lowpass and bandpass cases. If yes, decrease the order
% by 2 until the criteria are just met
%-------------------------------------------------------
if firtyp==1 | firtyp==3
```

```
N=NS+1;
  ll=0;
    while ll < 1
    N=N-2;
    [h,wind]=winsubad(firtyp,itype,N,wp,ws,dp,ds);
%-------------------------------------------------------
% Test whether the criteria at are met at the edges.
%-------------------------------------------------------
    isu=0;
if firtyp < 3
  [a,z]=zeroam(h,ws,ws,1);
  if abs(a) > ds isu=1;end
  [a,z]=zeroam(h,wp,wp,1);
  if abs(1-a) > dp isu=1;end
end
if firtyp > 2
  [a1,z]=zeroam(h,ws(1),ws(1),1);
  [a2,z]=zeroam(h,ws(2),ws(2),1);
  if max(abs(a1),abs(a2)) > ds isu=1;end
  [a1,z]=zeroam(h,wp(1),wp(1),1);
  [a2,z]=zeroam(h,wp(2),wp(2),1);
  if max(abs(1-a1),abs(1-a2)) > dp isu=1;end
end
    ll=1;
    if isu==0 ll=0;
    hs=h;winds=wind;NS=N;end
    end
end
end
fprintf('Minimum length is %g.\',NS)
%-------------------------------------------------------
% Plot the responses.
%-------------------------------------------------------
figure(1)
subplot(211)
[H,W]=zeroam(winds,.0,1.,8*1024);
plot(W/pi,20*log10(abs(H)));
amin=2.5*max(20*log10(abs(H(4*1024:8*1024+1))));
amax=1.2*max(20*log10(abs(H(1:1024))));
grid;axis([0 1 amin amax])
title('Window function');
ylabel('Amplitude in dB');
xlabel('Angular frequency omega/pi')
subplot(212)
impz(winds); grid;
title('Window function');xlabel('n in samples');
```

```
ylabel('Impulse response');title('Window function');
figure(2)
subplot(211)
[H,W]=zeroam(hs,.0,1.,8*1024);
amm=20*log10(min(dp,ds))*5/3;
plot(W/pi,20*log10(abs(H)));axis([0 1 amm 20]);grid;
title('Resulting filter');
ylabel('Amplitude in dB');
xlabel('Angular frequency omega/pi');
subplot(212)
impz(hs);
title('Window function');xlabel('n in samples');
ylabel('Impulse response');title('Resulting filter');
grid
figure(3)
if firtyp==1
  x1=0;x2=wp;
end
if firtyp==2
  x1=wp;x2=1;
end
if firtyp==3
  x1=wp(1);x2=wp(2);
end
if firtyp==4
  x1=0;x2=1;
end
subplot(211)
dpp=dp;
if dp >= 1.5*ds dpp=1.5*ds; end
plot(W/pi,H);axis([x1 x2 1-dpp 1+dpp]);
grid;
title('Passband details for the resulting filter');
ylabel('Zero-phase response');
xlabel('Angular frequency omega/pi');
if firtyp==1
  x1=ws;x2=1;
end
if firtyp==2
  x1=0;x2=ws;
end
if firtyp==3
  x1=0;x2=1;
end
if firtyp==4
  x1=ws(1);x2=ws(2);
```

```
end
subplot(212)
dss=ds;
if ds >= 1.5*dp dss=1.5*dp; end
plot(W/pi,H);axis([x1 x2 -dss dss]);
grid;
title('Stopband details for the resulting filter');
ylabel('Zero-phase response');
xlabel('Angular frequency omega/pi');
save fircoe hs -ascii -double
disp(' ')
disp('For further use, you can find')
disp('the filter coefficients in fircoe')
 ************************************************************************************************
function [h,wind]=winsubad(firtyp,itype,N,wp,ws,dpp,dss)
%-------------------------------------------------------
% Finds the parameters for the Kaiser (itype=1),
% Saramäki (itype=2), the Dolph-Chebyshev (itype=3),
% and the transitional window (itype=4) such that the
% the stopband or passband ripple criterion of the
% resulting filter is just met with 0.1 % accuracy.
% The other one is less than the given value.
% dpp is the maximum deviation from unity in the
% passband(s) and dss is the maximum deviation from zero
% in the stopband(s).
% If N is too low, the above is not possible.
% winwinad.m takes care of this and increases N.
%-------------------------------------------------------
% firtyp=1,2,3,4 for lowpass, highpass, bandpass,
% and bandstop filters
%-------------------------------------------------------
% The program returns both the filter coefficient h and
% the window function w. N is the length; the
% corresponding order is N-1.
%-------------------------------------------------------

%-------------------------------------------------------
% Tapio Saramäki  16.10.97
%-------------------------------------------------------
% This can be found in SUN's: ~ts/matlab/dsp/winsubad.m
%-------------------------------------------------------
if firtyp < 3
  wpp=wp;wss=ws;wcc=(wp+ws)/2;
  if wp > ws;
    wpp=ws;
    wss=wp;
```

```
  end
end
if firtyp > 2
  wpp=wp(1);wss=ws(1);
  if wp(1) > ws(1);
    wpp=ws(1);
    wss=wp(1);
  end
  if abs(ws(2)-wp(2)) < abs(ws(1)-wp(1))
    wpp=wp(2);
    wss=ws(2);
    if wp(2) > ws(2);
      wpp=ws(2);
      wss=wp(2);
    end
  end
end
%-----------------------------------------------------------
ds=dss;
dp=dpp;
accuracy=1;
while accuracy == 1
  if itype==1
    [N1,alpha,wc]=kaiord(wpp,wss,dp,ds);
    wind=rot90(kaiser(N,alpha));
  end
  if itype==2
    [N1,beta,wc]=sarord(wpp,wss,dp,ds);
    %wind=saramaki(N,beta);
    wind=transit(N,beta,1); %faster
  end
  if itype==3
    [N1,beta,wc]=dchord(wpp,wss,dp,ds);
    %wind=dcheb(N,beta);
    wind=transit(N,beta,0); %faster
  end
  if itype==4
    [N1,beta,rho,wc]=traord(wpp,wss,dp,ds);
    wind=transit(N,beta,rho);
  end
  if firtyp > 2 wc=(wp+ws)/2;end
  if firtyp==1
    h=firwind(N-1,wcc,wind);
    [H,W]=freqz(h,1,8*1024);
    na=round(ws*8*1024)+500/N;
    nb=round(wp*8*1024)-500/N;
```

```
  amaxa=max(abs(H(na:8*1024)));
  amaxb=max(abs(1-abs(H(1:nb))));
end
if firtyp==2
  h=firwind(N-1,wcc,wind,'high');
  [H,W]=freqz(h,1,8*1024);
  na=round(ws*8*1024)-500/N;
  nb=round(wp*8*1024)+500/N;
  amaxa=max(abs(H(1:na)));
  amaxb=max(abs(1-abs(H(nb:8*1024))));
end
if firtyp==3
  h=firwind(N-1,wc,wind);
  [H,W]=freqz(h,1,8*1024);
  na1=round(ws(1)*8*1024)-500/N;
  na2=round(ws(2)*8*1024)+500/N;
  amaxx(1)=max(abs(H(1:na1)));
  amaxx(2)=max(abs(H(na2:8*1024)));
  amaxa=max(amaxx);
  na1=round(wp(1)*8*1024)+500/N;
  na2=round(wp(2)*8*1024)-500/N;
  amaxb=max(abs(1-abs(H(na1:na2))));
end
if firtyp==4
  h=firwind(N-1,wc,wind,'stop');
  [H,W]=freqz(h,1,8*1024);
  na1=round(ws(1)*8*1024)+500/N;
  na2=round(ws(2)*8*1024)-500/N;
  amaxa=max(abs(H(na1:na2)));
  na1=round(wp(1)*8*1024)-500/N;
  na2=round(wp(2)*8*1024)+500/N;
  amaxx(1)=max(abs(1-abs(H(1:na1))));
  amaxx(2)=max(abs(1-abs(H(na2:8*1024))));
  amaxb=max(amaxx);
end
nda=-20*log10(dss)+20*log10(amaxa);
nda=-20*log10(ds)+nda;
ds=10^(-nda/20);
ndb=-20*log10(dpp)+20*log10(amaxb);
ndb=-20*log10(dp)+ndb;
dp=10^(-ndb/20);
if abs((amaxa-dss)/dss) < .0001 & amaxb < dpp
  accuracy=0;
end
if (abs(amaxb-dpp)/dpp) < .0001 & amaxa < dss
  accuracy=0;
```

```
  end
  if dp < dpp/10 & ds < dss/10
    accuracy=0; % N is too low
  end
end
*************************************************************************************f
unction [h] = firwind(N,wc,wind,type)
%-----------------------------------------
% h=firwind(N,wc,wind,type) is a program for
% designing linear-phase FIR filters of order
% N using the window method.
% It return the filter  coefficients in the
% length N+1 vector h.
%-----------------------------------------
% For lowpass, highpass, bandpass, and bandpass
% cases, wc contains the cutoff frequencies as a
% fraction of pi (half the sampling rate).
% For lowpass and highpass cases, wc is a scalar.
% For bandpass and bandstop cases, wc= [wc1 wc2]
% with wc1 and wc2 being the two cutoff points.
% For differentiators, the ideal response is
% omega up to pi*wc and zero for the this cutoff
% up to pi. For Hilbert transformers, wc may take
% any arbitrary value.
% type indicates the filter type. If absent, the
% program designs a lowpass or a bandpass filter
% depending on whether there is one or two elements
% in wc.
% h=firwind(N,wc,wind), h=firwind(N,wc,wind,'high'),
% h=firwind(N,wc,wind), h=firwind(N,wc,wind 'stop'),
% h=firwind(N,wc,wind,'differentiator'), and
% h=firwind(N,wc,wind,'Hilbert') design an N'th order
% lowpass, highpass, bandpass, differentiator, and
% Hilbert transformer filter, respectively, using the
% length N+1 window with coefficients given in wind.
%-------------------------------------------------
% The available windows are:
%
% Fixed windows:
%
% - rectangular
% - Bartlett
% - Hann
% - Hamming
% - Blackman
%
```

```
% Adjustable windows:
%
% - Kaiser
% - Saramäki
% - Dolph-Chebyshev
% - Transitional
%-------------------------------------------------------

%-------------------------------------------------------
% Tapio Saramäki  1.11.97
%-------------------------------------------------------
% This can be found in SUN's: ~ts/matlab/dsp/firwind.m
%-------------------------------------------------------
if length(wind)~=N+1
   error('The window and filter lengths must be the same')
end
if nargin == 3
   type = 'con';
end
nodd=rem(N,2);
%-------------------------------------------------------
% lowpass case
%-------------------------------------------------------
if type(1)=='c' & length(wc)==1
%-------------------------------------------------------
% check
%-------------------------------------------------------
  if wc > 1 | wc < 0
    error('Cutoff point must fall between 0 and 1.')
  end
%-------------------------------------------------------
  if nodd==0
     n1=-N/2:1:-1;
     n2=1:1:N/2;
     h1=sin(pi*wc*n1)./(pi*n1);
     h2=sin(pi*wc*n2)./(pi*n2);
     h=[h1 wc h2];
  end
  if nodd==1
    n=-N/2:1:N/2;
    h=sin(pi*wc*n)./(pi*n);
  end
end
%-------------------------------------------------------
% highpass case
%-------------------------------------------------------
```

```
if (type(1:3)=='hig' | type(1:3)=='Hig') & length(wc)==1
%-------------------------------------------------------
% check
%-------------------------------------------------------
  if wc > 1 | wc < 0
    error('Cutoff point must fall between 0 and 1.')
  end
  if nodd==1
    error('For the highpass case, order must be even')
  end
%-------------------------------------------------------
  n1=-N/2:1:-1;
  n2=1:1:N/2;
  h1=-sin(pi*wc*n1)./(pi*n1);
  h2=-sin(pi*wc*n2)./(pi*n2);
  h=[h1 1-wc h2];
end
%-------------------------------------------------------
% bandpass case
%-------------------------------------------------------
if type(1)=='c' & length(wc)==2
%-------------------------------------------------------
% check
%-------------------------------------------------------
  if wc(1) > 1 | wc(1) < 0 | wc(2) > 1 | wc(2) < 0
    error('Cutoff points must fall between 0 and 1.')
  end
  if wc(1) > wc(2)
    error('Lower cutoff must be less than upper cutoff.')
  end
%-------------------------------------------------------
  if nodd==0
     n1=-N/2:1:-1;
     n2=1:1:N/2;
     h1=(sin(pi*wc(2)*n1)-sin(pi*wc(1)*n1))./(pi*n1);
     h2=(sin(pi*wc(2)*n2)-sin(pi*wc(1)*n2))./(pi*n2);
     h=[h1 wc(2)-wc(1) h2];
  end
  if nodd==1
    n=-N/2:1:N/2;
    h=(sin(pi*wc(2)*n)-sin(pi*wc(1)*n))./(pi*n);
  end
end
%-------------------------------------------------------
% bandstop case
%-------------------------------------------------------
```

```
if (type(1)=='s' | type(1)=='S') & length(wc)==2
%-------------------------------------------------------
% check
%-------------------------------------------------------
  if wc > 1 | wc < 0
    error('Cutoff point must fall between 0 and 1.')
  end
  if nodd==1
    error('For the bandstop case, order must be even')
  end
  if nodd==1
    error('For the bandstop case, order must be even')
  end
%-------------------------------------------------------
     n1=-N/2:1:-1;
     n2=1:1:N/2;
     h1=(sin(pi*wc(1)*n1)-sin(pi*wc(2)*n1))./(pi*n1);
     h2=(sin(pi*wc(1)*n2)-sin(pi*wc(2)*n2))./(pi*n2);
     h=[h1 1-wc(2)+wc(1) h2];
end
%-------------------------------------------------------
% Hilbert transformer
%-------------------------------------------------------
if (type(1:3)=='hil' | type(1:3)=='Hil')
%-------------------------------------------------------
  if nodd==0
    n=-N/2:1:-1;
    h1=-(1-cos(pi*n))./(pi*n);
    h2=-rot90(rot90(h1));
    h=[h1 0 h2];
  end
  if nodd==1
    n=-N/2:1:-1/2;
    h1=-ones(size(n))./(pi*n);
    h2=-rot90(rot90(h1));
    h=[h1 h2];
  end
end
%-------------------------------------------------------
% Differentiator
%-------------------------------------------------------
if (type(1)=='d' | type(1)=='D')
%-------------------------------------------------------
  if nodd==0
    n=-N/2:1:-1;
    h1=-(1/pi)*(sin(n*pi*wc)./(n.^2)-pi*wc*cos(n*pi*wc)./n);
```

```
   h2=-rot90(rot90(h1));
   h=[h1 0 h2];
  end
  if nodd==1
   n=-N/2:1:-1/2;
   h1=-(4/pi)*(sin(n*pi*wc)./((2*n).^2)-(pi*wc/2)*cos(n*pi*wc)./(2*n));
   h2=-rot90(rot90(h1));
   h=[h1 h2];
  end
end
%old for the case wc=1
%  if nodd==0
%   n=-N/2:1:-1;
%   h1=cos(pi*n)./n;
%   h2=-rot90(rot90(h1));
%   h=[h1 0 h2];
%  end
%  if nodd==1
%   n=-N/2:1:-1/2;
%   h1=cos(pi*(n+1/2))./(pi*n.^2);
%   h2=-rot90(rot90(h1));
%   h=[h1 h2];
%  end
%-----------------------------------------
% Final result
%-----------------------------------------
h=h.*wind;
*****************************************************************************************f
unction [N,alpha,wc]=kaiord(wp,ws,dp,ds)
%---------------------------------------
% Given the passband and stopband edges,
% wp and ws as a fraction of pi (half the
% sampling rate, and the passband and stop-
% band ripples, dp and ds, for a lowpass
% FIR filter, [N,alpha,wc]=kaiord(wp,ws,dp,ds)
% evaluates the estimated length, N, and
% the estimated value of the parameter alpha
% for the Kaiser window. Also the cutoff
% frequency of the ideal filter wc=(wp+ws)/2
% is calculated.
%-----------------------------------------

%-----------------------------------------
% N=ceil(2*M+1) where M is given in Table 4-5 in
% T. Saramäki, "Finite impulse response Filter
% Design" in Handbook for Digital Signal Processing,
```

```
% S. K. Mitra and J. F. Kaiser, Eds, John Wiley &
% Sons, 1993
% Also alpha is evaluated according to the formula
% in the same table.
% Programmed by Tapio Saramäki, 1995
% This can be found in SUN's:
% ~ts/matlab/dsp/kaiord.m
%------------------------------------------------
%cutoff frequency of the ideal filter
wc=(wp+ws)/2;
%------------------------------------------------
%determine alpha
%------------------------------------------------
d=dp;
if ds < d d=ds;end
As=-20*log10(d);
alpha=0.5842*(As-21)^0.4+0.07886*(As-21);
if As>50 alpha=0.1102*(As-8.7); end
%------------------------------------------------
%determine N
%------------------------------------------------
N=ceil((As-7.95)/(7.18*(ws-wp))+1);
%------------------------------------------------
**********************************************************************************************f
unction [N,beta,wc]=sarord(wp,ws,dp,ds)
%--------------------------------------
% Given the passband and stopband edges,
% wp and ws as a fraction of pi (half the
% sampling rate, and the passband and stop-
% band ripples, dp and ds, for a lowpass
% FIR filter, [N,beta,wc]=sarord(wp,ws,dp,ds)
% evaluates the estimated length, N, and
% the estimated value of the parameter beta
% for the Saramäki window. Also the cutoff
% frequency of the ideal filter wc=(wp+ws)/2
% is calculated.
%--------------------------------------

%------------------------------------------------
% N=round(2*M+1) where M is given in Table 4-5 in
% T. Saramaki, "Finite impulse response Filter
% Design" in Handbook for Digital Signal Processing,
% S. K. Mitra and J. F. Kaiser, Eds, John Wiley &
% Sons, 1993
% Also beta is evaluated according to the formula
% in the same table.
```

```
% Programmed by Tapio Saramäki, 1995
% This can be found in SUN's
% ~ts/matlab/dsp/kaiord.m
%-----------------------------------------------
%cutoff frequency of the ideal filter
%-----------------------------------------------
wc=(wp+ws)/2;
%-----------------------------------------------
%determine beta
%-----------------------------------------------
d=dp;
if ds < d d=ds;end
As=-20*log10(d);
beta=0.000121*(As-21)^2+0.0224*(As-21)+1;
if As>65 beta=0.033*As+0.062; end
if As>110 beta=0.345*As-0.097; end
%-----------------------------------------------
%determine N
%-----------------------------------------------
N=ceil((As-8.15)/(7.18*(ws-wp))+1);
%-------------------------------------------------
**************************************************************************************************f
unction [w]=saramaki(N,beta)
%-----------------------------------------------
% Evaluates the impulse response of the
% Saramäki window of length N for the given
% value of beta.
%-----------------------------------------------

%-----------------------------------------------
% For N odd, equations (4.49b) to (4.52) in
% T. Saramäki, "Finite impulse response Filter
% Design" in Handbook for Digital Signal Processing,
% S. K. Mitra and J. F. Kaiser, Eds, John Wiley &
% Sons, 1993 are directly applied with M=(N-1)/2.
% The impulse response coefficients are given
% by starting with index n=1.
% For N even, M=N-1 and finally every second impulse
% response coefficient (occuring at even n) is
% disregarded, as explained on page 177 in the
% above-mentioned chapter
% Programmed by Tapio Saramäki, 1995
% This can be found in SUN's
% ~ts/matlab/dsp/saramaki.m
%-------------------------------------------------
if 2*floor(N/2)==N itype=2; %N even
```

```
else itype=1; %N odd
end
if itype==1 M=(N-1)/2;
else M=N-1; end
%Evaluate gamma, equation (4.49b),
gamma=beta*2*pi/(2*M+1);
gamma=(1+cos(2*pi/(2*M+1)))/(1+cos(gamma));
%-----------------------------------------------------
% Evaluate gamma1=gamma/2 and gamma2=gamma-1
%-----------------------------------------------------
gamma1=gamma/2;
gamma2=gamma-1;
%-----------------------------------------------------
% Initialize working vectors vectors.
% First, impulse response values for the zero-phase
% window is evaluated from 0 to M, in practice
% from n=1 to n=M+1;
% Based on these evaluations, a causal windows is
% then generated
%-----------------------------------------------------
w1(1,2*M+1)=0; w2(1,M+2)=0; w3(1,M+2)=0;
ww(1,M+2)=0; w(1,N)=0;
w1(1)=1; w2(1)=gamma2; w2(2)=gamma1;
ww(1)=w1(1)+2*w2(1); ww(2)=2*w2(2);
for i=2:M
     w3(1)=2*gamma1*(w2(2)+w2(2))+2*gamma2*w2(1)-w1(1);
     ww(1)=ww(1)+2*w3(1);
   for k=2:i+1
     w3(k)=2*gamma1*(w2(k-1)+w2(k+1))+2*gamma2*w2(k)-w1(k);
     ww(k)=ww(k)+2*w3(k); end
   for k=1:M+1 w1(k)=w2(k); w2(k)=w3(k); end
end
for i=1:M+1 w1(i)=ww(M+2-i)/ww(1); end
for i=2:M+1 w1(i+M)=ww(i)/ww(1); end
%-----------------------------------------------------------
% For N even, every second impulse response value is
% disregarded
%-----------------------------------------------------------
if itype==1 nn=1; else nn=2; end
for i=1:N w(i)=w1(1+(i-1)*nn); end
%-----------------------------------------------------------
**************************************************************************************************f
unction [N,beta,wc]=dchord(wp,ws,dp,ds)
%-----------------------------------------
% Given the passband and stopband edges,
% wp and ws as a fraction of pi (half the
```

```
% sampling rate, and the passband and stop-
% band ripples, dp and ds, for a lowpass
% FIR filter, [N,beta,wc]=dchord(wp,ws,dp,ds)
% evaluates the estimated length, N, and
% the estimated value of the parameter beta
% for the Dolph-Chebyshev window. Also
% the cutoff frequency of the ideal filter
% wc=(wp+ws)/2 is calculated.
%-------------------------------------------

%-------------------------------------------------
% N=ceil(2*M+1) where M is given in Table 4-5 in
% T. Saramäki, "Finite impulse response Filter
% Design" in Handbook for Digital Signal Processing,
% S. K. Mitra and J. F. Kaiser, Eds, John Wiley &
% Sons, 1993
% Also beta is evaluated according to the formula
% in the same table.
% Programmed by Tapio Saramäki, 1995
% This can be found in SUN's:
% ~ts/matlab/dsp/dchord.m
%-------------------------------------------------
%cutoff frequency of the ideal filter
%-------------------------------------------------
wc=(wp+ws)/2;
%-------------------------------------------------
%determine beta
%-------------------------------------------------
d=dp;
if ds < d d=ds;end
As=-20*log10(d);
beta=0.0000769*(As)^2+0.0248*As+0.330;
if As>60 beta=0.0000769*(As)^2+0.0248*As+0.330; end
%-------------------------------------------------
%determine N
%-------------------------------------------------
N=ceil((1.028*As-8.4)/(7.18*(ws-wp))+1);
%-------------------------------------------------------
**********************************************************************************************f
unction [w]=dcheb(N,beta)
%-------------------------------------------------
% [w]=dcheb(N,beta) evaluates the impulse response
% of the Dolph-Chebyshev window of length N for
% the given value of beta.
%-------------------------------------------------
```

```
%-------------------------------------------------
% For N odd, equations (4.53) and (4.54) in
% T. Saramaki, "Finite impulse response Filter
% Design in Handbook for Digital Signal Processing,
% S. K. Mitra and J. F. Kaiser, Eds, John Wiley &
% Sons, 1993 are directly applied with M=(N-1)/2.
% The impulse response coefficients are given
% by starting with index n=1.
% For N even, M=N-1 and finally every second impulse
% response coefficient (occuring at even n) is
% disregarded, as explained on page 177 in the
% above-mentioned chapter.
% Programmed by Tapio Saramaki, 1995.
% This can be found in SUN's
% ~ts/matlab/dsp/dcheb.m
%-------------------------------------------------
if 2*floor(N/2)==N itype=2; %N even
else itype=1; %N odd
end
if itype==1 M=(N-1)/2;
else M=N-1; end
%-------------------------------------------------------
% Evaluate gamma, equation (4.49b)
%-------------------------------------------------------
gamma=beta*2*pi/(2*M+1);
gamma=(1+cos(pi/(2*M)))/(1+cos(gamma));
%-------------------------------------------------------
%Evaluate gamma1=gamma/2 and gamma2=gamma-1
%-------------------------------------------------------
gamma1=gamma/2;
gamma2=gamma-1;
%-------------------------------------------------------
%Initialize working vectors.
%First, impulse response values for the zero-phase
%window is evaluated from 0 to M, in practice
%from n=1 to n=M+1;
%Based on these evaluations, a causal windows is
%then generated
%-------------------------------------------------------
w1(1,2*M+1)=0; w2(1,M+2)=0; w3(1,M+2)=0; w(1,N)=0;
w1(1)=1; w2(1)=gamma2; w2(2)=gamma1;
for i=2:M
     w3(1)=2*gamma1*(w2(2)+w2(2))+2*gamma2*w2(1)-w1(1);
   for k=2:i+1
     w3(k)=2*gamma1*(w2(k-1)+w2(k+1))+2*gamma2*w2(k)-w1(k);
     end
```

```
   for k=1:M+1 w1(k)=w2(k); w2(k)=w3(k); end
end
for i=1:M+1 w1(i)=w3(M+2-i)/w3(1); end
for i=2:M+1 w1(i+M)=w3(i)/w3(1); end
%-------------------------------------------------------
%For N even, every second impulse response value is
%disregarded
%-------------------------------------------------------
if itype==1 nn=1; else nn=2; end
for i=1:N w(i)=w1(1+(i-1)*nn); end
%-------------------------------------------------------
*************************************************************************************f
unction [N,beta,rho,wc]=traord(wp,ws,dp,ds)
%-----------------------------------------
% Given the passband and stopband edges,
% wp and ws as a fraction of pi (half the
% sampling rate, and the passband and stop-
% band ripples, dp and ds, for a lowpass
% FIR filter, this program
% evaluates the estimated length, N, and
% the estimated values of the parameters alpha
% and rho for the transitional window. Also
% the cutoff frequency of the ideal filter
% wc=(wp+ws)/2 is calculated.
%-----------------------------------------

%-----------------------------------------
% N=ceil(2*M+1) where M is given in Table 4-5 in
% T. Saramaki, "Finite impulse response Filter
% Design" in Handbook for Digital Signal Processing,
% S. K. Mitra and J. F. Kaiser, Eds, John Wiley &
% Sons, 1993
% Also beta is evaluated according to the formula
% in the same table.
% Programmed by Tapio Saramaki, 1995
% This can be found in SUN's:
% ~ts/matlab/dsp/traord.m
%-----------------------------------------------
%
%cutoff frequency of the ideal filter
wc=(wp+ws)/2;
%
%determine beta and rho
d=dp;
if ds < d d=ds;end
As=-20*log10(d);
```

```
beta=0.000154*(As)^2+0.0153*As+.465;
if As>60 beta=0.0000204*(As)^2+0.0303*As+.032; end
rho=0.4;
if As>50 rho=0.5; end
if As>75 rho=0.6; end
%
%determine N
N=ceil((0.00036*As^2+0.951*As-9.4)/(7.18*(ws-wp))+1);
%-----------------------------------------------------
**********************************************************************************************f
unction [w]=transit(N,beta,rho)
%-----------------------------------------------
% Evaluates the impulse response of the transitional
% window of length N for the given values of beta
% and rho.
%-----------------------------------------------

%-----------------------------------------------
% For N odd, the unscaled frequency response is
% generated accoring to Equation (4.55) in
% T. Saramaki, "Finite impulse response Filter
% Design" in Handbook for Digital Signal Processing,
% S. K. Mitra and J. F. Kaiser, Eds, John Wiley &
% Sons, 1993 are directly applied with M=(N-1)/2.
% The impulse response coefficients are given
% by starting with index n=1.
% For N even, M=N-1 and finally every second impulse
% response coefficient (occuring at even n) is
% disregarded, as explained on page 177 in the
% above-mentioned article.
% The time domain response is generated using IFFT.
% Tapio Saramäki October 31, 1997
% This can be found in SUN's
% ~ts/matlab/dsp/transit.m
%-----------------------------------------------
% Note: transi.m is an older version
%-----------------------------------------------
if 2*floor(N/2)==N
  itype=2; %N even
else
  itype=1; %N odd
end
if itype==1
  M=(N-1)/2;
else
  M=N-1;
```

```
end
%-------------------------------------------------------
% The causal window is generated using the IFFT.
% To generate a causal window, we evaluate the zero-phase
% frequency response of our window at 2^I > 2M+1 equally
% spaced frequencies and use the IFFT.
%-------------------------------------------------------
I=log2(2*M+1);I=floor(I)+1;k=2^I;
omega=0:2*pi/k:2*(k-1)*pi/k;
a1=cos(pi/(2*M+1));
a2=cos(pi/(4*M));
b1=cos(beta*pi/(2*M+1))/a1;
b2=cos(beta*pi/(2*M+1))/a2;
A=ones(size(omega));
for k=1:M
  om1=(b1)*cos(k*pi/(2*M+1));
  om2=(b2)*cos((2*k-1)*pi/(4*M));
  cc=rho*acos(om1)+(1-rho)*acos(om2);
  A=A.*(cos(omega)-cos(2*cc));
end
b=ifft(A);
b=real(b);
b=fftshift(b);
for k=1:2*M+1
  what(k)=b(2^(I-1)+1-(M+1)+k);
end
what=what/what(M+1);
%For N even, every second impulse response value is
%disregarded
if itype==1
  nn=1;
else
  nn=2;
end
for i=1:N
  w(i)=what(1+(i-1)*nn);
end
%-------------------------------------------------
**************************************************************************************************f
unction [A,w]=zeroam(h,u1,u2,M)
%evaluates the zero-phase frequency response
%of a linear phase FIR filter at points
%w=u1*pi:(u2-u1)*pi/M:u2*pi
%Tapio Saramaki 28.10.1995
N=length(h);
NN=floor((N-1)/2);
```

```
iodd=0;
if 2*round(N/2)==N iodd=1; end
isy=1;
sum1=sum(h(1:NN));sum2=sum(h(N+1-NN:N));
if abs(sum2-sum1)/abs(sum1) < .5 isy=0; end
%if abs(h(N)-h(1))/abs(h(1)) < .5 isy=0; end
if iodd==0 & isy==0 itype=1;end
if iodd==1 & isy==0 itype=2;end
if iodd==0 & isy==1 itype=3;end
if iodd==1 & isy==1 itype=4;end
L=floor((N+2)/2);
if M > 1 w=u1*pi:(u2-u1)*pi/M:u2*pi;
end
if M==1 w=u1*pi; end
A=zeros(size(w));
if itype==1 A=A+h(L);end
for k=1:L-1
  if itype==1
  A=A+2*h(L-k)*cos(k*w);end
  if itype==2
  A=A+2*h(L-k)*cos((k-.5)*w);end
  if itype==3
  A=A+2*h(L-k)*sin(k*w);end
  if itype==4
  A=A+2*h(L-k)*sin((k-.5)*w);end
end
**********************************************************************************************
```